MAKE SMILES

A Project Report Submitted
In Partial Fulfillment of the Requirements
for the Degree of

MASTER OF COMPUTER APPLICATIONS

by
PriyankaTyagi
(1900290149074)

Under the Supervision of

Mr. AnkitVerma
KIET Group of Institutions, Ghaziabad

to the

FACULTY OF MCA

DR. APJ ABDUL KALAM TECHNICAL UNIVERSITY
(Formerly Uttar Pradesh Technical University) LUCKNOW

July 2021

DECLARATION

Iherebydeclarethattheworkpresentedinthisreportentitled“MAKE SMILES”, was carried out by me. I have not submitted the matter embodied in this report for the award of any other degree or diploma of any other University or Institute.

Ihavegivenduecredittotheoriginalauthors/sourcesforallthewords,ideas,diagrams, graphics,computerprograms,experiments,results,thatarenotmyoriginalcontribution. I have used quotation marks to identify verbatim sentences and given credit to the originalauthors/sources.

I affirm that no portion of my work is plagiarized, and the experiments and results reportedinthereportarenotmanipulated.Intheeventofacomplaintofplagiarismand the manipulation of the experiments and results, I shall be fully responsible and answerable.

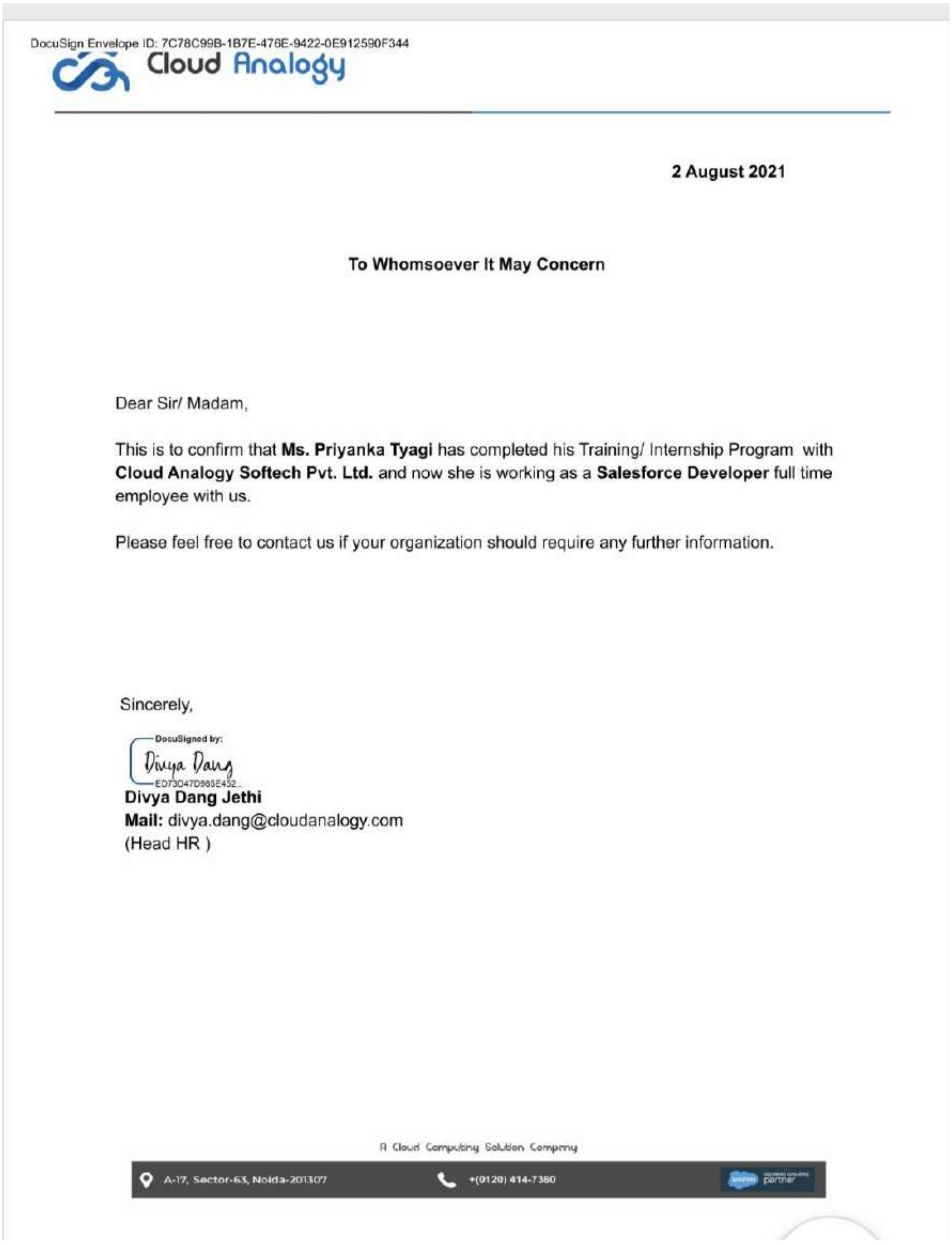
Name	: Priyanka Tyagi
Roll No.	: 1900290149074
Branch	: Master of ComputerApplications

Priyanka Tyagi

(Candidate Signature)

Date:

TRAINING CERTIFICATE



DocuSign Envelope ID: 7C78C99B-1B7E-476E-9422-0E912590F344

Cloud Analogy

2 August 2021

To Whomsoever It May Concern

Dear Sir/ Madam,

This is to confirm that **Ms. Priyanka Tyagi** has completed his Training/ Internship Program with **Cloud Analogy Softech Pvt. Ltd.** and now she is working as a **Salesforce Developer** full time employee with us.

Please feel free to contact us if your organization should require any further information.

Sincerely,

DocuSigned by:
Divya Dang
ED73D47D985E452...

Divya Dang Jethi
Mail: divya.dang@cloudanalogy.com
(Head HR)

A Cloud Computing Solution Company

A-17, Sector-63, Noida-201307 +(0120) 414-7360 partner

CERTIFICATE

Certified that **Priyanka Tyagi** (**1900290149074**) has carried out the project work presented in this report entitled "**MAKE SMILES**" for the award of **Master of Computer Application** from Dr. A.P.J. Abdul Kalam Technical University, Lucknowundermysupervision.Thereportembodiesresultoforiginalwork,andstudies arecarriedoutbythestudenthimselfandthecontentsofthereportdonotformthebasis for the award of any other degree to the candidate or to anybody else from this or any otherUniversity.

Mr. Ankit Verma **External Examiner**

Associate Professor

Dept. of Computer Applications

KIET Group of Institutions, Ghaziabad

Dr. Ajay Kumar Srivastava

Professor & Head

Department of Computer Applications

KIET Group of Institutions, Ghaziabad

Date:

MAKE SMILES

PRIYANKA TYAGI

ABSTRACT

Cloud Computing is a crowd/group of unknown resources that are giving for a specific purpose to the user. This CRM is intended for a particular client (Organization) i.e., MAKE SMILES to stored their current manual, paper-based system. The proposed new structure is to control the Wisher's information. These facilities are to be provided in a well-organized, cost effective manner, with the objective of reducing the time and resources currently required for such tasks.

An important part of the process it provides comfort and joy to the young angels during their time here. This information usually involves; Wishers individual information and medicinal history, family information, wisher school information another necessary information related to an organization's properties may be effectually utilized HMS will automate the administration of the hospital making it more well-organized and errorfree.

All the service industries in modern world are highly dependent on the quality of the data, defining objectives from available data (historical data) and utilizing the same to achieve those objectives. Our mission here was to understand and gather the requirement for an enterprise level quality dashboard for a hospital client, so that the business and executives get man overall understanding of the Hospital management growth to help them take high level decisions based on the data represented on this Dashboard.

ACKNOWLEDGEMENT

Successinlifeisneverattainedsinglehandedly.MydeepestgratitudegoestomyProject supervisor**.** **Mr. Ankit Verma, Associate Professor, Department of Computer Applications** for his guidance, help and encouragement throughout my research work. Their enlightening ideas, comments, and suggestions. Words are not enough to express my gratitude for his insightful comments and administrative help at variousoccasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directlyand indirectly provided me moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Priyanka Tyagi

1900290149074

Table of Contents

List of Figures

Chapter – 1

INTRODUCTION

PROJECTDESCRIPTION

For a huge organization like MAKE SMILES which holds the data of many diverse fields, it is a backbreaker for the higher personnel to monitor petty level tasks to transaction level tasks. Heedlessness in monitoring the progress and funds usage is depreciating organizations discipline and advancementThere should be bridge of trust between hospital and customer.

This CRM is intended for a particular client (Organization) i.e., MAKE SMILES to stored their current manual, paper-based system. The proposed new structure is to control the Wishers information. These facilities are to be provided in a well-organized, cost effective manner, with the objective of reducing the time and resources currently required for such tasks.

An important part of the process it provides comfort and joy to the young angels during their time here. This information usually involves; Wishers individual information and medicinal history, family information, wisher school information another necessary properties may be effectually utilized HMS will automate the administration of the hospital making it more well-organized and errorfree.

PROJECTSCOPE

The recent rapid increase in the amount of medical information has pushed hospitals to confront an essential issue which is how to utilize healthcare information technology to improvehealthcareservicesquality.Customerrelationshipmanagementsystem(CRMS) isaninnovativetechnologywhichfacilitatestheprocesstoacquire,develop,andmaintain customer relationships more efficiently and effectively. Customer relationship management (CRM) for healthcare providers is an approach to learn all they can about their customers and prospects, to communicate relevant, timely information to them,and totrackresultstomakeprogramadjustmentsnecessary.Therapidincreaseintheamount of medical information has pushed hospitals to confront a critical issue, which is how to utilize information technologies to manage large amounts of customer information and then improve the quality of customer services. The adoption of a customer relationship managementsystem(CRMS)thusisincreasedgloballyamonghospitals.Thepercentage of hospitals which utilize Web sites for sales and marketing purposes has increased 2.47 timesfrom1995(17%)to2000(59%)intheUS.Also,customersatisfactionismeasured by healthcare providers as a key factor of strategy and an important determinant of long-term feasibility and success under competitive situation. In addition, maintaining and increasing customer loyalty level is essential for any service company’s long-term success. Creating a conceptual framework for the CRM health care system using multivariatemeasurementsystemsuchasfactoranalysisorprincipalcomponentanalysis causes improvement in business by increasing the level of customer satisfaction and customer loyalty.

1.3 HARDWARE/SOFTWARE USED IN THIS PROJECT

1.3.1 Introduction

Information technology (IT) is the use of any computers, storage, networking and other physical devices, infrastructure and processes to create, process, store, secure and exchange all forms of electronic data. Typically, IT is used in the context of business operations, as opposed to technology used for personal or entertainment purposes. The commercial use of IT encompasses both computer technology and telecommunications. [14]

As this is a cloud-based application so there is no need of sophisticated hardware only there is a need of Browser application and Internet connection. Cloud provider provides the development infrastructure to create application.

A cloud application, or cloud app, is a software program where cloud-based and local components work together. This model relies on remote servers for processing logic that is accessed through a web browser with a continual internet connection.

Cloud development involves, well, developing the cloud. ... This involves developing the cloud architecture such as planning, organizing, and designing to implementing and structuring cloud delivery models (Iaas, Paas, Iaas). Additional tasks in cloud development include managing the cloud service delivery models.[3]

Cloud application servers typically are located in a remote data center operated by a third-party cloud services infrastructure provider. Cloud-based application tasks may encompass email, file storage and sharing, order entry, inventory management, word processing, customer relationship management (CRM), data collection, or financial accounting features.

There are many CRM software choices available with a wide range of features and add-on integrations. While it is great to have so many options, it can be difficult to determine which CRM software is the best fit for your business. The key is to examine your goals and objectives for the project and how these align with your overall business goals; determine what your user base needs the software to do; assess what features are most important to your business; evaluate the integration potential of the software; make sure that any solution you select is mobile-friendly; and resist the urge to build a custom solution. While this process takes time and resources, a thorough evaluation will help you select a system that yields maximum ROI.[2]

1.3.2 Benefits of cloud apps

Fast response to business needs. Cloud applications can be updated, tested and deployed quickly, providing enterprises with fast time to market and agility. This speed can lead to culture shifts in business operations.

Simplified operation. Infrastructure management can be outsourced to third-party cloud providers.

Instant scalability. As demand rises or falls, available capacity can be adjusted.

API use. Third-party data sources and storage services can be accessed with an application programming interface (API). Cloud applications can be kept smaller by using APIs to hand data to applications or API-based back-end services for processing or analytics computations, with the results handed back to the cloud application. Vetted APIs impose passive consistency that can speed development and yield predictable results.

Gradual adoption. Refactoring legacy, on-premises applications to a cloud architecture in steps, allows components to be implemented on a gradual basis.

Reduced costs. The size and scale of data centers run by major cloud infrastructure and service providers, along with competition among providers, has led to lower prices. Cloud-based applications can be less expensive to operate and maintain than equivalents on-premises installation.

Improved data sharing and security. Data stored on cloud services is instantly available to authorized users. Due to their massive scale, cloud providers can hire world-class security experts and implement infrastructure security measures that typically only large enterprises can obtain. Centralized data managed by IT operations personnel is more easily backed up on a regular schedule and restored should disaster recovery become necessary.

Cloud computing strategies will undoubtedly continue to be a huge part of every organization’s network strategy in the coming years. With scalable power and flexible services, cloud providers give companies the tools they need to drive better business results. Data centers have a key role to play in building these infrastructures, making it critical that IT professionals keep a close eye on the latest developments in these interconnected industries.[1]

cloud application, or cloud app, is a software program where cloud-based and local components work together. This model relies on remote servers for processing logic that is accessed through a web browser with a continual internet connection.

Cloud application servers typically are located in a remote data center operated by a third-party cloud services infrastructure provider. Cloud-based application tasks may encompass email, file storage and sharing, order entry, inventory management, word processing, customer relationship management (CRM), data collection, or financial accounting features.

Fast response to business needs: Cloud applications can be updated, tested and deployed quickly, providing enterprises with fast time to market and agility. This speed can lead to culture shifts in business operations.

Simplified operation: Infrastructure management can be outsourced to third-party cloud providers.

Instant scalability: As demand rises or falls, available capacity can be adjusted.

API use: Third-party data sources and storage services can be accessed with an application programming interface (API). Cloud applications can be kept smaller by using APIs to hand data to applications or API-based back-end services for processing or analytics computations, with the results handed back to the cloud application. Vetted APIs impose passive consistency that can speed development and yield predictable results.

Gradual adoption: Refactoring legacy, on-premises applications to a cloud architecture in steps, allows components to be implemented on a gradual basis.

Reduced costs: The size and scale of data centers run by major cloud infrastructure and service providers, along with competition among providers, has led to lower prices.

Cloud-based applications can be less expensive to operate and maintain than equivalents on-premises installation.

Improved data sharing and security: Data stored on cloud services is instantly available to authorized users. Due to their massive scale, cloud providers can hire world-class security experts and implement infrastructure security measures that typically only large enterprises can obtain. Centralized data managed by IT operations personnel is more easily backed up on a regular schedule and restored should disaster recovery become necessary.

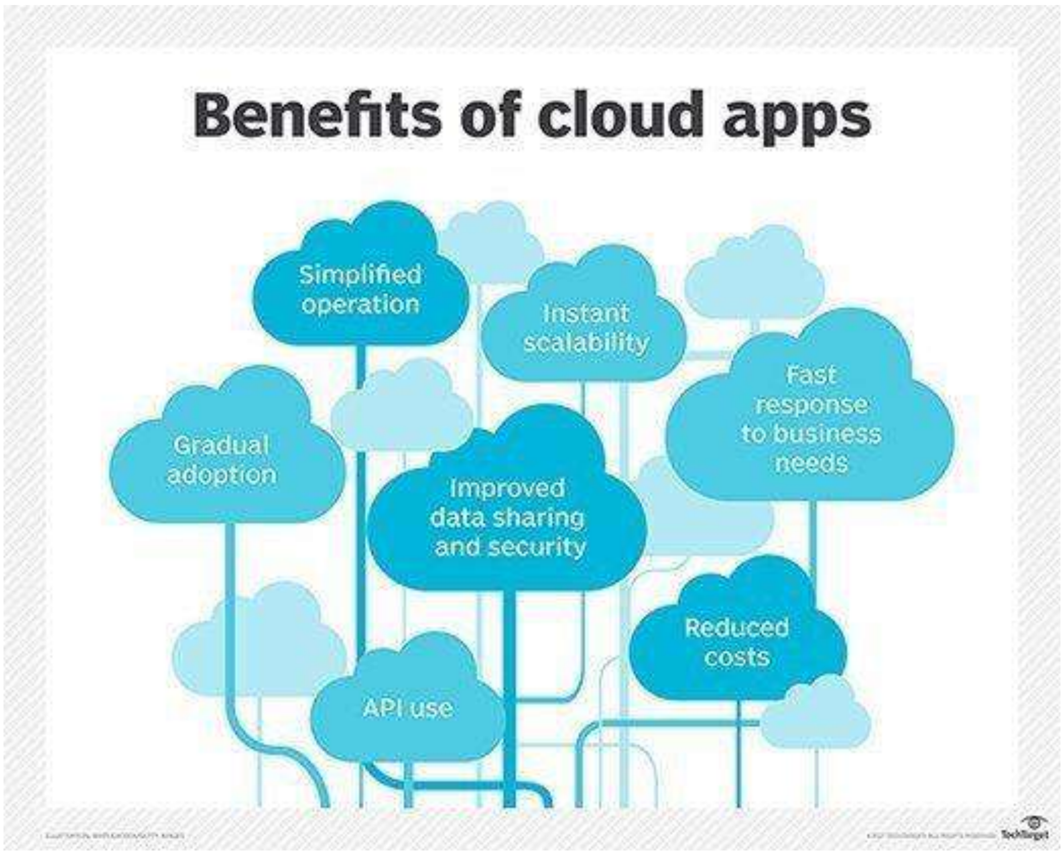


Fig 1.1 Benefits of Clouds

1.3.3 How cloud apps work

Data is stored and compute cycles occur in a remote data center typically operated by a third-party company. A back end ensures uptime, security and integration and supports multiple access methods.

Cloud applications provide quick responsiveness and don't need to permanently reside on the local device. They can function offline, but can be updated online.

While under constant control, cloud applications don't always consume storage space on a computer or communications device. Assuming a reasonably fast internet connection, a well-written cloud application offers all the interactivity of a desktop application, along with the portability of a web application.

With the advancement of remote computing technology, clear lines between cloud and web applications have blurred. The term cloud application has gained great cachet, sometimes leading application vendors with any online aspect to brand them as cloud applications.

Cloud and web applications access data residing on distant storage. Both use server processing power that may be located on premises or in a distant data center.

A key difference between cloud and web applications is architecture. A web application or web-based application must have a continuous internet connection to function. Conversely, a cloud application or cloud-based application performs processing tasks on a local computer or workstation. An internet connection is required primarily for downloading or uploading data.

A web application is unusable if the remote server is unavailable. If the remote server becomes unavailable in a cloud application, the software installed on the local user device can still operate, although it cannot upload and download data until service at the remote server is restored.

The difference between cloud and web applications can be illustrated with two common productivity tools, email and word processing. Gmail, for example, is a web application that requires only a browser and internet connection. Through the browser, it's possible to open, write and organize messages using search and sort capabilities. All processing logic occurs on the servers of the service provider (Google, in this example) via either the internet's HTTP or HTTPS protocols.

A CRM application accessed through a browser under a fee-based software as a service (SaaS) arrangement is a web application. Online banking and daily crossword puzzles are also considered web applications that don't install software locally.

An example of a word-processing cloud application that is installed on a workstation is Word's Microsoft Office 365. The application performs tasks locally on a machine without an internet connection. The cloud aspect comes into play when users save work to an Office 365 cloud server.

Chapter – 2

Feasibility Study

A feasibility study is a preliminary study which investigates the information of prospective users and determines the resources requirements, costs, benefits and feasibility of proposed system. A feasibility study takes into account various constraints within which the system should be implemented and operated. In this stage, the resource needed for the implementation such as computing equipment, manpower and costs are estimated. The estimated are compared with available resources and a cost benefit analysis of the system is made. The feasibility analysis activity involves the analysis of the problem and collection of all relevant information relating to the project. The main objectives of the feasibility study are to determine whether the project would be feasible in terms of economic feasibility, technical feasibility and operational feasibility and schedule feasibility or not. It is to make sure that the input data which are required for the project are available. Thus, we evaluated the feasibility of the system in terms of the following categories:

- **Technical feasibility**
- **Operational feasibility**
- **Economical feasibility**

2.1 Technical Feasibility:

Evaluating the technical feasibility is the trickiest part of a feasibility study. This is because, at the point in time there is no any detailed designed of the system, making it difficult to access issues like performance, costs (on account of the kind of technology to be deployed) etc. A number of issues have to be considered while doing a technical analysis; understand the different technologies involved in the proposed system. Before commencing the project, we have to be very clear about what are the technologies that are to be required for the development of the new system. Is the required technology available? HR’s register is technically feasible.

2.2 Operational Feasibility:

Operational feasibility is the measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development.

The operational feasibility assessment focuses on the degree to which the proposed development project fits in with the existing business environment and objectives with regard to development schedule, delivery date, corporate culture and existing business processes.

To ensure success, desired operational outcomes must be imparted during design and development. These include such design-dependent parameters as reliability, maintainability, supportability, usability, producibility, disposability, sustainability, affordability and others. These parameters are required to be considered at the early stages of design if desired operational behaviours are to be realised. A system design and development require appropriate and timely application of engineering and management efforts to meet the previously mentioned parameters. A system may serve its intended purpose most effectively when its technical and operating characteristics are engineered into the design. Therefore, operational feasibility is a critical aspect of systems engineering that needs to be an integral part of the early design phases We have created the project which is operational feasible too. And this project will meet the needs by completing the project.

2.3 Economical Feasibility:

Our project is economical feasible. Economic feasibility attempts to weigh the costs of developing and implementing a new system, against the benefits that would accrue from having the new system in place. This feasibility study gives the top management the economic justification for the new system. A simple economic analysis which gives the actual comparison of costs and benefits are much more meaningful in this case. In addition, this proves to be useful point of reference to compare actual costs as the project progresses. There could be various types of intangible benefits on account of automation. These could increase customer satisfaction, improvement in product quality, better decision making, and timeliness of information, expediting activities, improved accuracy of operations, better documentation and record keeping, faster retrieval of information, better employee morale.

Chapter – 3

Design& Coding

3.1 Design Data Model

Schema Builder provides a dynamic environment for viewing and modifying all the objects and relationships in your app. This greatly simplifies the task of designing, implementing, and modifying your data model, or schema. Schema Builder is enabled by default. We can view our existing schema and interactively add new custom objects, custom fields, and relationships, simply by dragging and dropping. Schema Builder automatically implements the changes and saves the layout of our schema any time you move an object. This eliminates the need to click from page to page to find the details of a relationship or to add a new custom field to an object in our schema. Schema Builder provides details like the field values, required fields, and how objects are related by displaying lookup and master-detail relationships. We can view the fields and relationships for both standard and custom objects.

Schema Builder lets you add the following to your schema:

- Custom objects
- Lookup relationships (through blue line)
- Master-detail relationships (through purple line)

- All custom fields except: Geolocation

Here is our project’s Data model created with the help of Schema Builder –

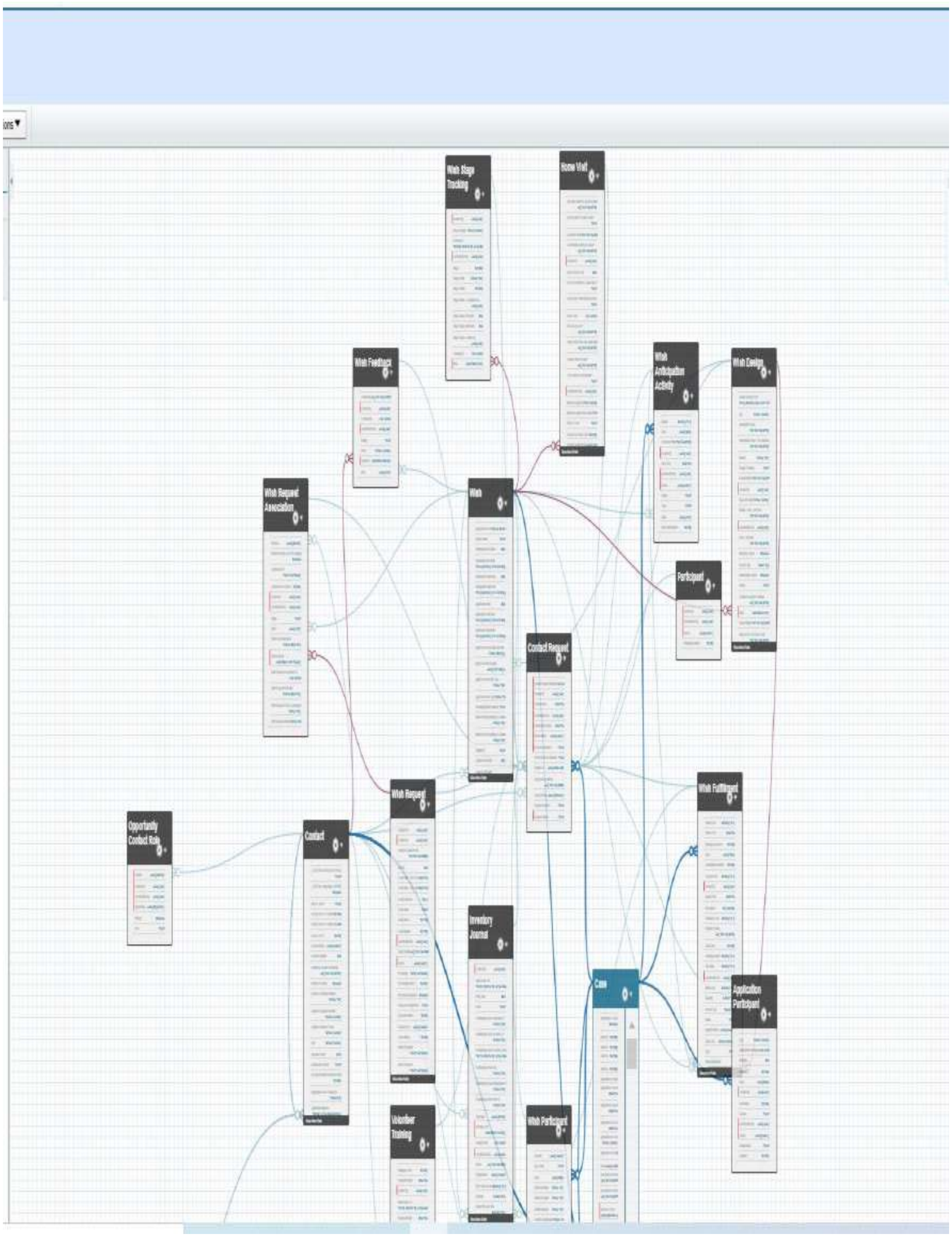
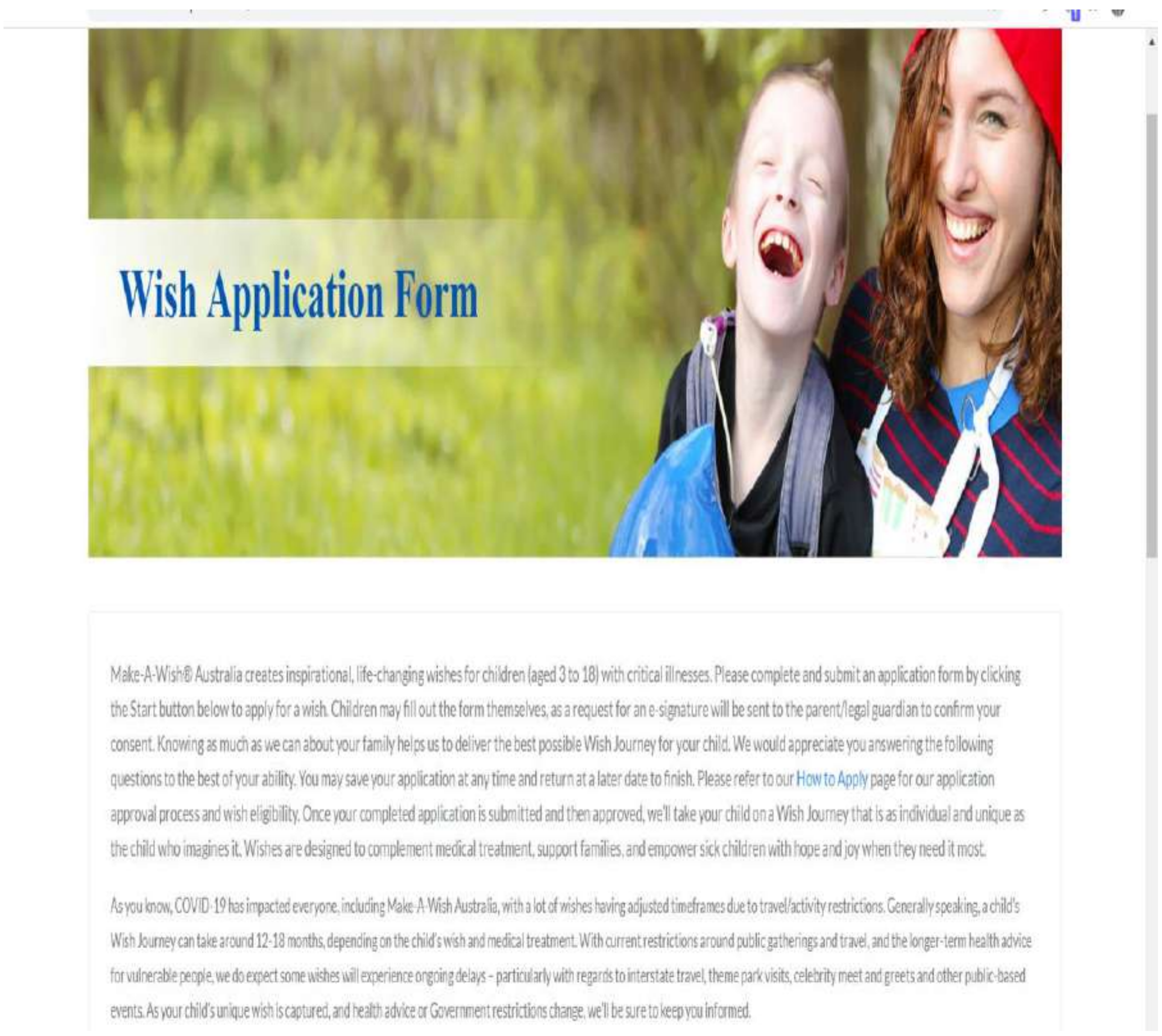


Fig 2.1 Data Model

3.2 Screenshots

3.2.1 Wish Form Page

Wish Form was developed on client's requirements through which operator can send any type of information like reminder of appointment, confirmation, cancellation, etc on patient's personal number, his family or on office number. This Page is developed in JavascriptHtml , CSS as it is light in weight and fast in accessing.



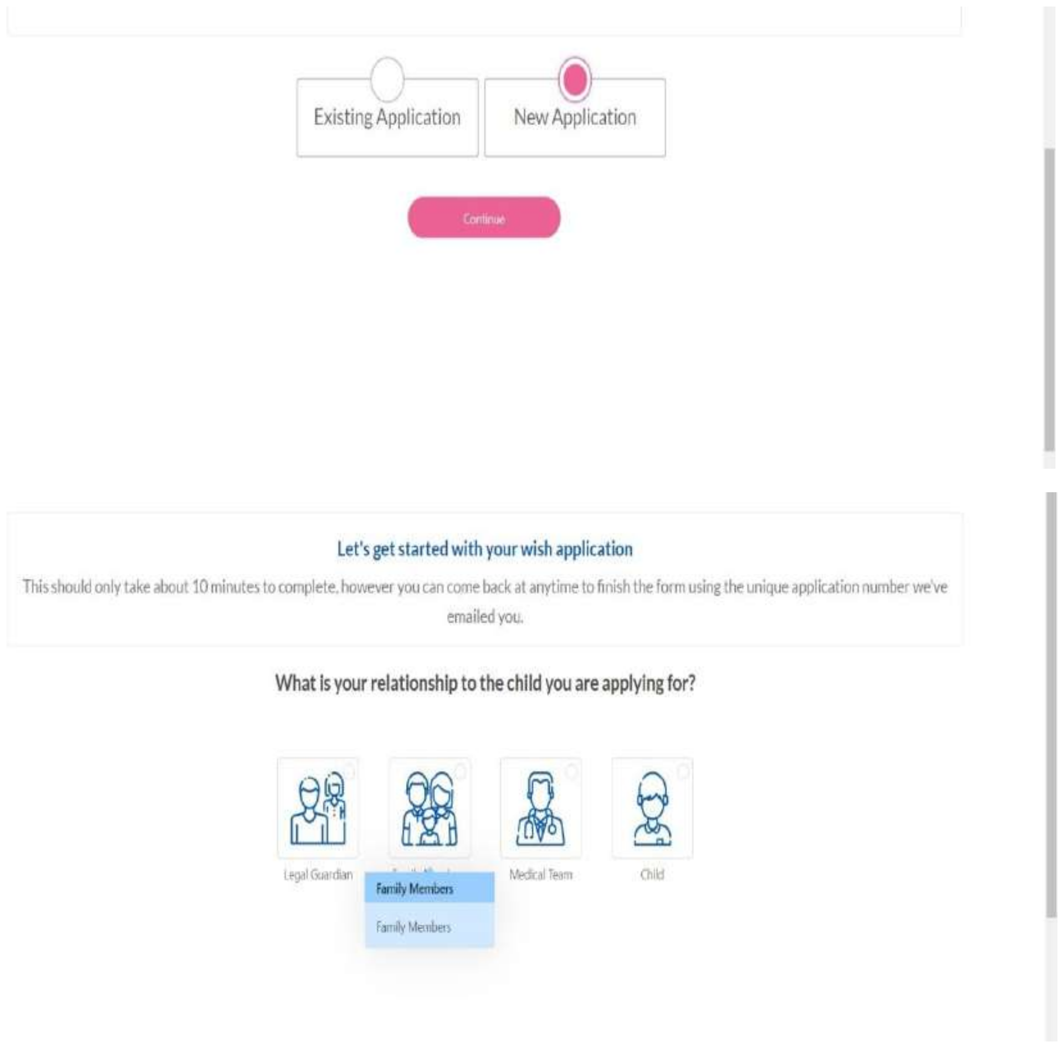
Make-A-Wish® Australia creates inspirational, life-changing wishes for children (aged 3 to 18) with critical illnesses. Please complete and submit an application form by clicking the Start button below to apply for a wish. Children may fill out the form themselves, as a request for an e-signature will be sent to the parent/legal guardian to confirm your consent. Knowing as much as we can about your family helps us to deliver the best possible Wish Journey for your child. We would appreciate you answering the following questions to the best of your ability. You may save your application at any time and return at a later date to finish. Please refer to our How to Apply page for our application approval process and wish eligibility. Once your completed application is submitted and then approved, we'll take your child on a Wish Journey that is as individual and unique as the child who imagines it. Wishes are designed to complement medical treatment, support families, and empower sick children with hope and joy when they need it most.

As you know, COVID-19 has impacted everyone, including Make-A-Wish Australia, with a lot of wishes having adjusted timeframes due to travel/activity restrictions. Generally speaking, a child's Wish Journey can take around 12-18 months, depending on the child's wish and medical treatment. With current restrictions around public gatherings and travel, and the longer-term health advice for vulnerable people, we do expect some wishes will experience ongoing delays – particularly with regards to interstate travel, theme park visits, celebrity meet and greets and other public-based events. As your child's unique wish is captured, and health advice or Government restrictions change, we'll be sure to keep you informed.

Fig 3.1 Wish Form

3.2.2 Application Page

This is the Application page from there a wisher can apply or any family member of child can apply for the wish by filling the wish form. We have two options one is for New Application and one is for Existing Application. The next page shown belown is used for the choosing relation of the candidate which fill the form. We can easily choose the relationship from the given relationship.



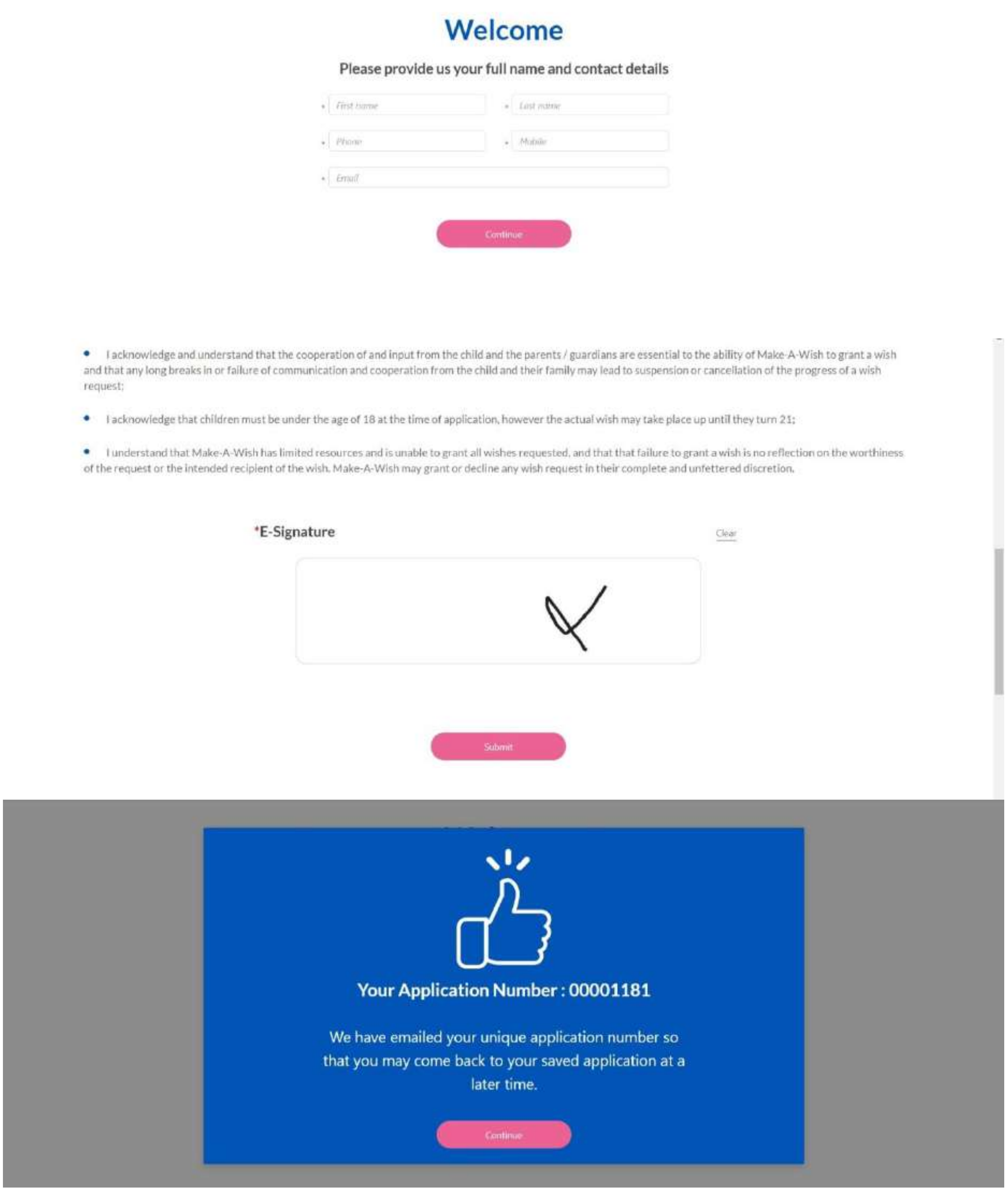
Let's get started with your wish application

This should only take about 10 minutes to complete, however you can come back at anytime to finish the form using the unique application number we've emailed you.

What is your relationship to the child you are applying for?

Fig 3.2. Application Page

3.2.3 Pages Related to the Form


Welcome

Please provide us your full name and contact details

- I acknowledge and understand that the cooperation of and input from the child and the parents / guardians are essential to the ability of Make-A-Wish to grant a wish and that any long breaks in or failure of communication and cooperation from the child and their family may lead to suspension or cancellation of the progress of a wish request;
- I acknowledge that children must be under the age of 18 at the time of application, however the actual wish may take place up until they turn 21;
- I understand that Make-A-Wish has limited resources and is unable to grant all wishes requested, and that that failure to grant a wish is no reflection on the worthiness of the request or the intended recipient of the wish. Make-A-Wish may grant or decline any wish request in their complete and unfettered discretion.

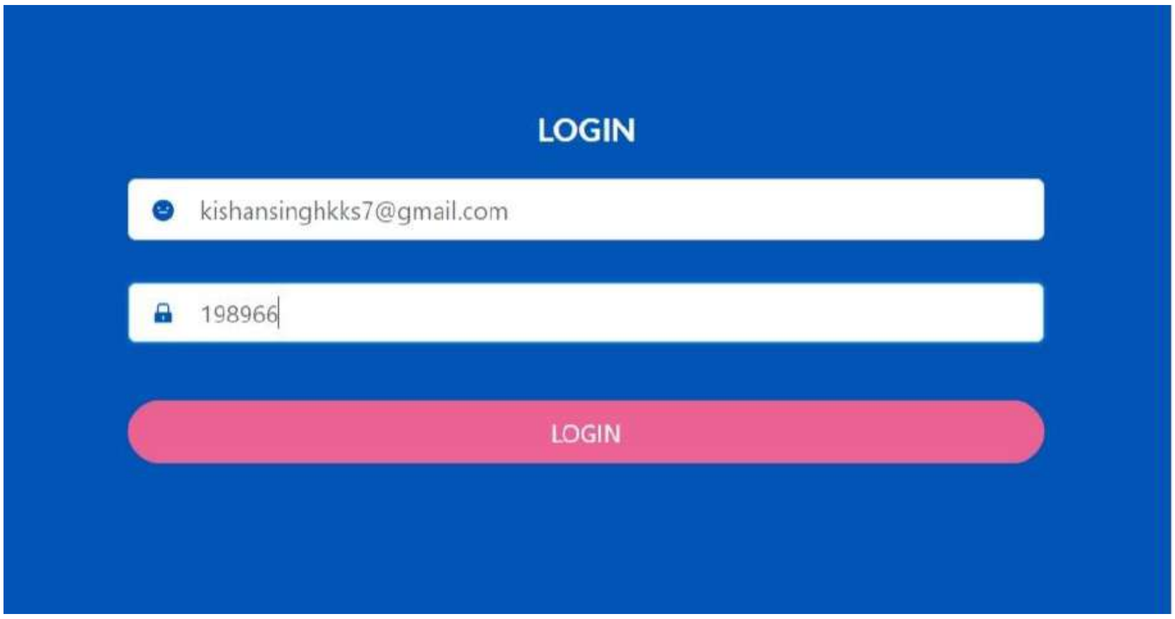
Your Application Number : 00001181

We have emailed your unique application number so that you may come back to your saved application at a later time.

Fig 3.3 **Pages Related to the Form**

3.2.4 Dashboard For Doctors And Social Workers

Dashboard component developed for the Doctors and the Social Workers, from the dashboard they can easily access the wish requests and also approve the request or may also create new wish application. We also verify the Doctors and Social Workers by sending the OTP on the required Email Ids.


LOGIN

kishansinghkks7@gmail.com

198966

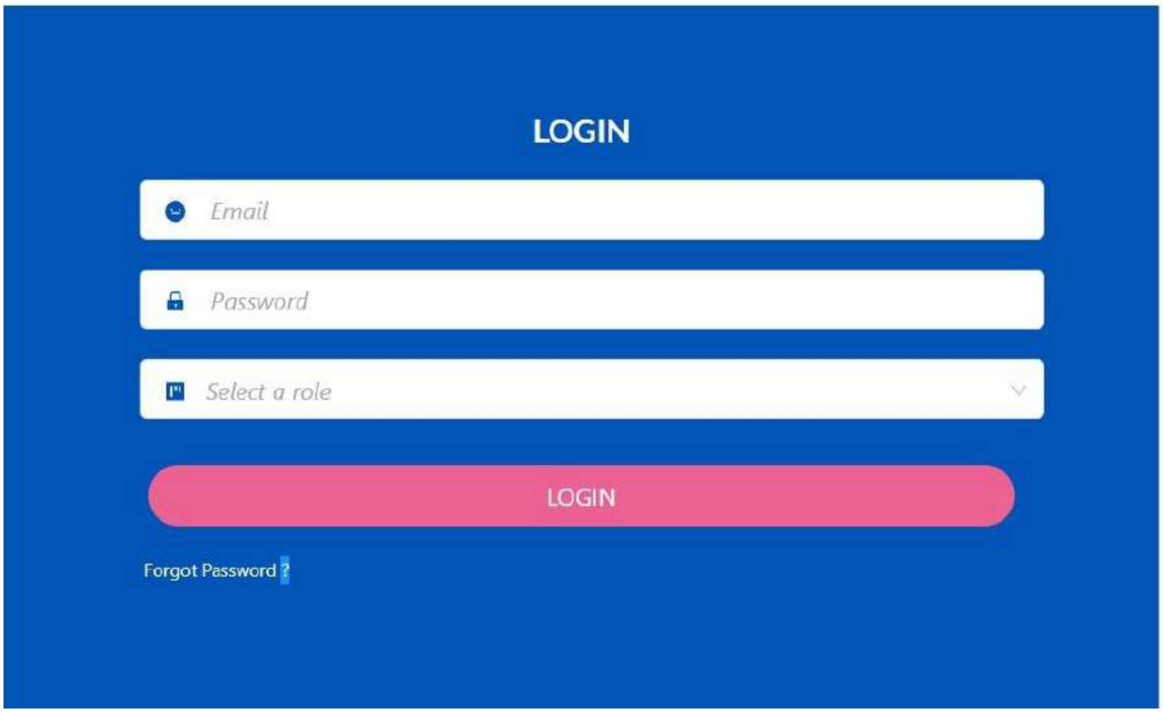
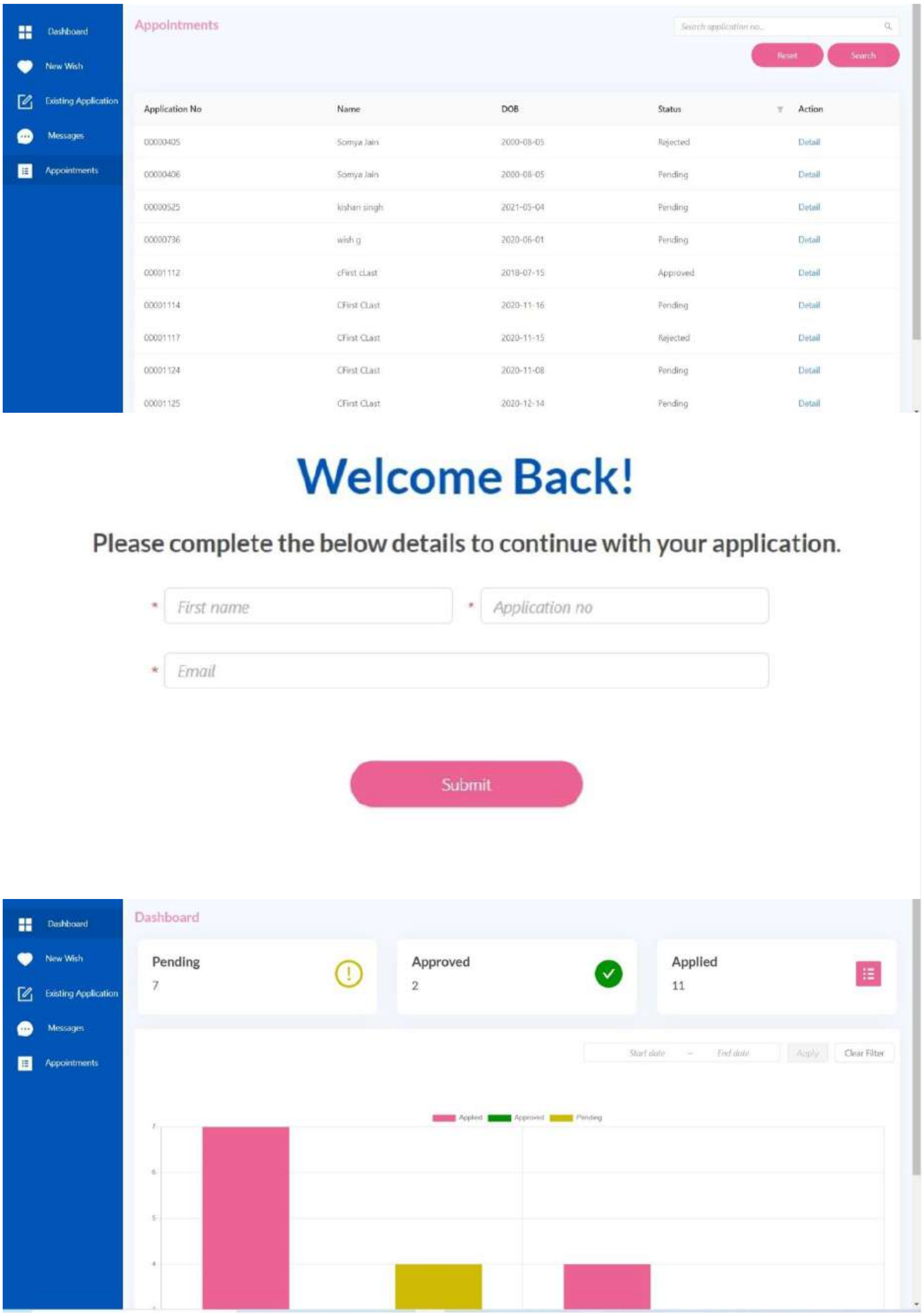

LOGIN

Fig 3.4 Dashboard Log In Pages



Appointments

Application No	Name	DOB	Status	Action
00000405	Somya Jain	2000-08-05	Rejected	Detail
00000406	Somya Jain	2000-08-05	Pending	Detail
00000525	kishan singh	2021-05-04	Pending	Detail
00000736	wish g	2020-06-01	Pending	Detail
00001112	cFirst cLast	2018-07-15	Approved	Detail
00001114	CFirst CLast	2020-11-16	Pending	Detail
00001117	CFirst CLast	2020-11-15	Rejected	Detail
00001124	CFirst CLast	2020-11-08	Pending	Detail
00001125	CFirst CLast	2020-12-14	Pending	Detail

Welcome Back!

Please complete the below details to continue with your application.

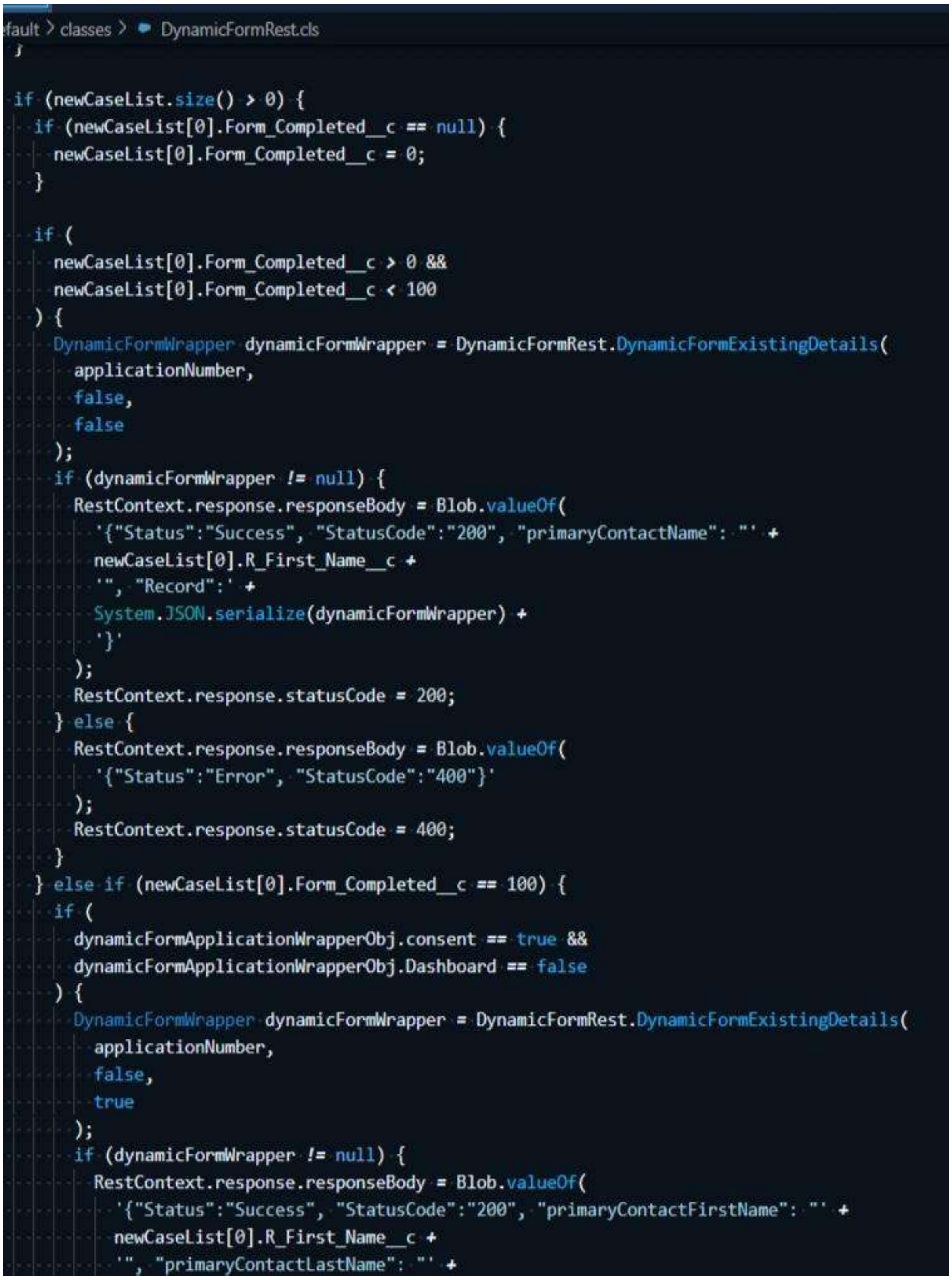
* First name
* Application no
* Email

Dashboard

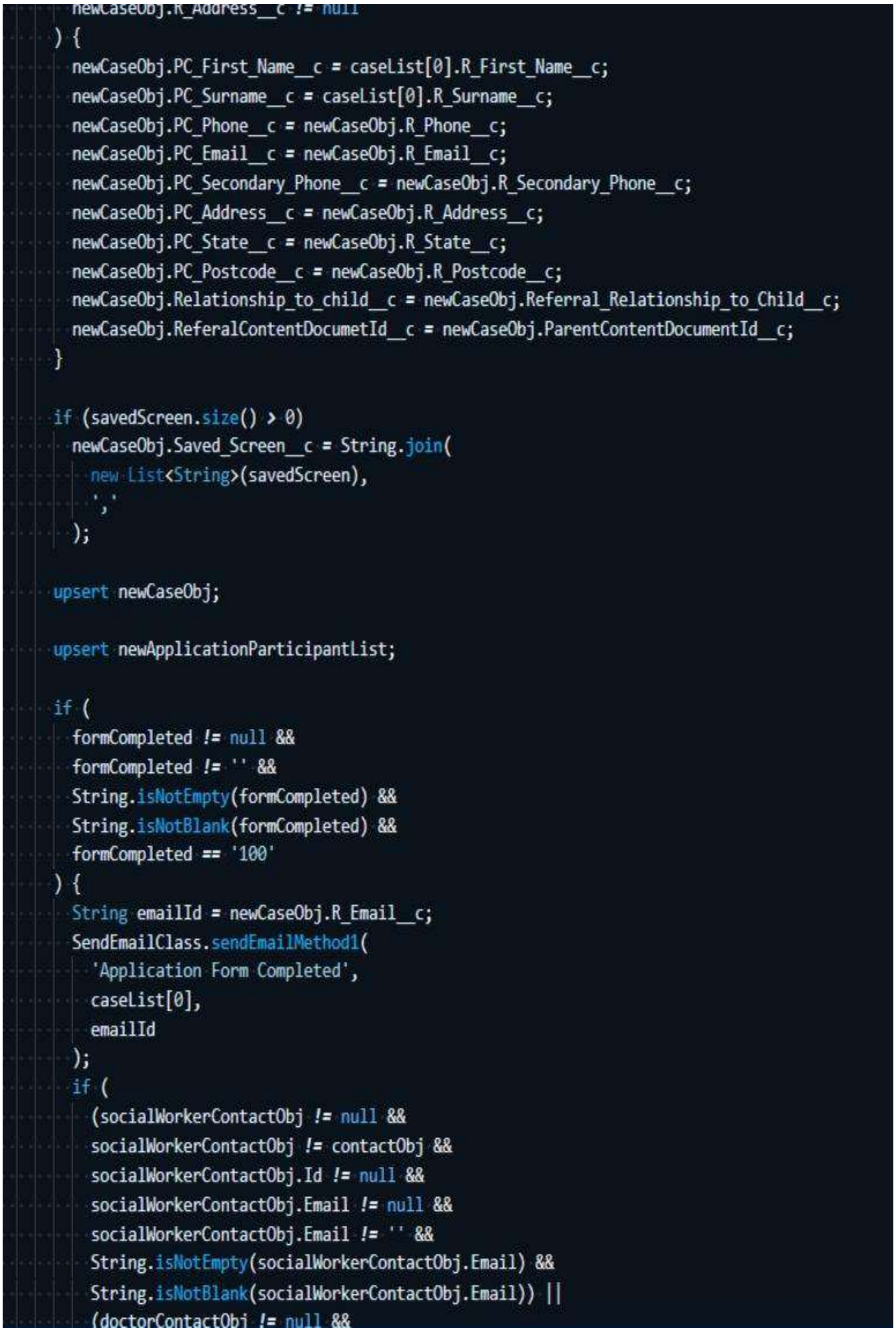
Pending 7
Approved 2
Applied 11

Fig 3.4 All the Pages attached in Dashboard

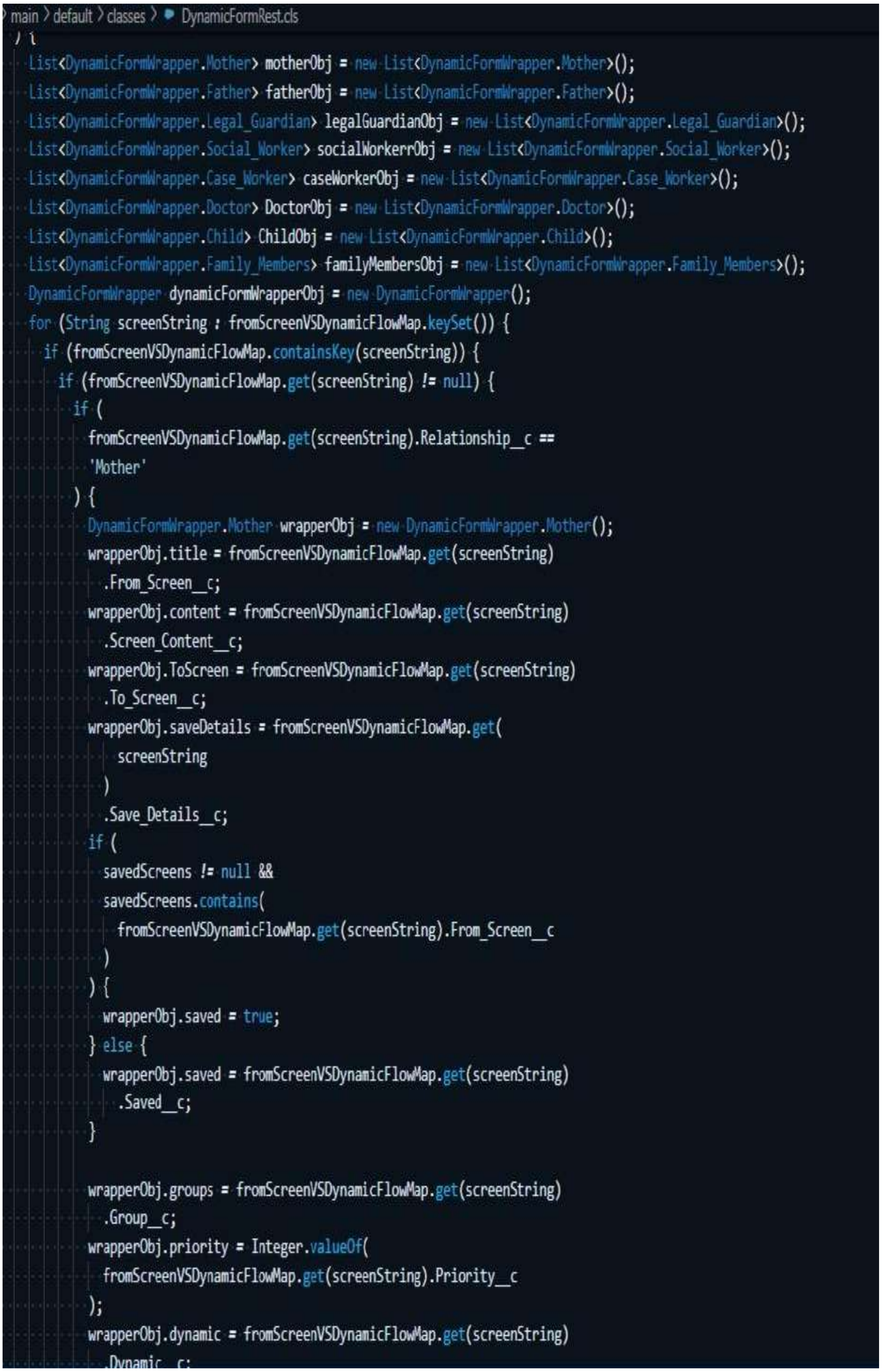
3.3 Source Code



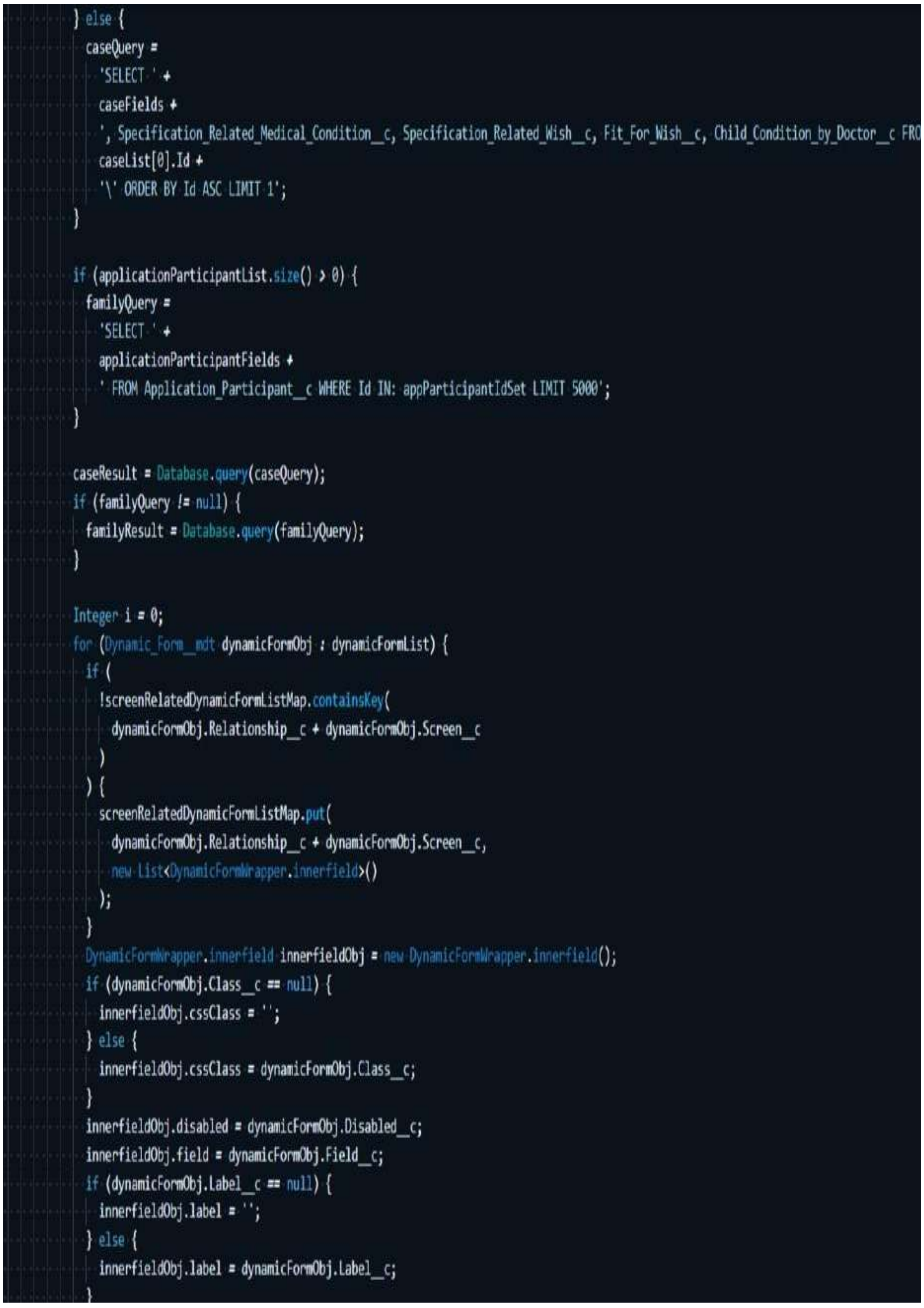
```
}

if (newCaseList.size() > 0) {
  if (newCaseList[0].Form_Completed__c == null) {
    newCaseList[0].Form_Completed__c = 0;
  }

  if (
    newCaseList[0].Form_Completed__c > 0 &&
    newCaseList[0].Form_Completed__c < 100
  ) {
    DynamicFormWrapper dynamicFormWrapper = DynamicFormRest.DynamicFormExistingDetails(
      applicationNumber,
      false,
      false
    );
    if (dynamicFormWrapper != null) {
      RestContext.response.responseBody = Blob.valueOf(
        '{"Status":"Success", "StatusCode":"200", "primaryContactName": "' +
        newCaseList[0].R_First_Name__c +
        '", "Record":' +
        System.JSON.serialize(dynamicFormWrapper) +
        '}'
      );
      RestContext.response.statusCode = 200;
    } else {
      RestContext.response.responseBody = Blob.valueOf(
        '{"Status":"Error", "StatusCode":"400"}'
      );
      RestContext.response.statusCode = 400;
    }
  } else if (newCaseList[0].Form_Completed__c == 100) {
    if (
      dynamicFormApplicationWrapperObj.consent == true &&
      dynamicFormApplicationWrapperObj.Dashboard == false
    ) {
      DynamicFormWrapper dynamicFormWrapper = DynamicFormRest.DynamicFormExistingDetails(
        applicationNumber,
        false,
        true
      );
      if (dynamicFormWrapper != null) {
        RestContext.response.responseBody = Blob.valueOf(
          '{"Status":"Success", "StatusCode":"200", "primaryContactFirstName": "' +
          newCaseList[0].R_First_Name__c +
          '", "primaryContactLastName": "' +
```



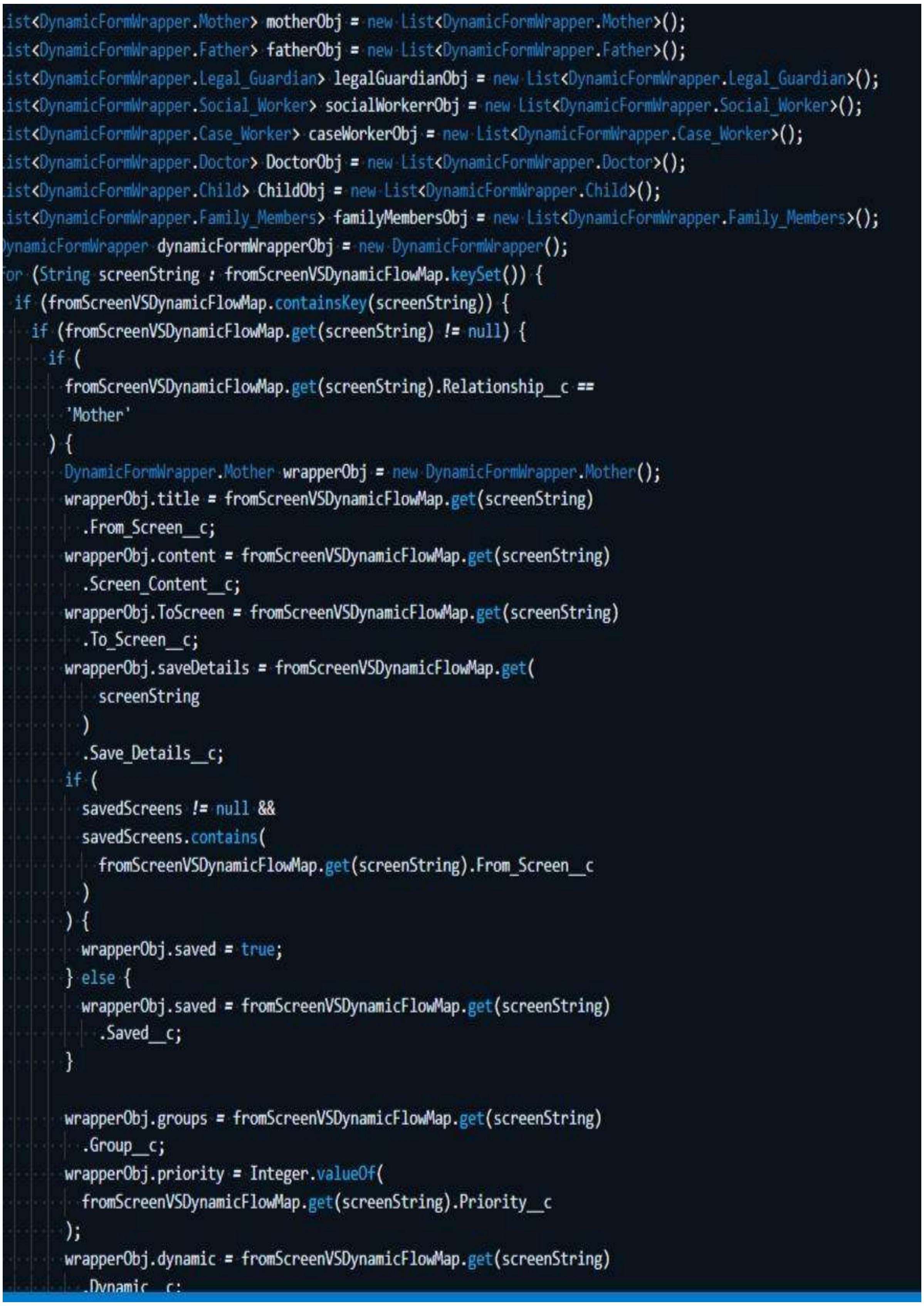
```
  newCaseObj.R_Address__c != null
) {
  newCaseObj.PC_First_Name__c = caseList[0].R_First_Name__c;
  newCaseObj.PC_Surname__c = caseList[0].R_Surname__c;
  newCaseObj.PC_Phone__c = newCaseObj.R_Phone__c;
  newCaseObj.PC_Email__c = newCaseObj.R_Email__c;
  newCaseObj.PC_Secondary_Phone__c = newCaseObj.R_Secondary_Phone__c;
  newCaseObj.PC_Address__c = newCaseObj.R_Address__c;
  newCaseObj.PC_State__c = newCaseObj.R_State__c;
  newCaseObj.PC_Postcode__c = newCaseObj.R_Postcode__c;
  newCaseObj.Relationship_to_child__c = newCaseObj.Referral_Relationship_to_Child__c;
  newCaseObj.ReferalContentDocumetId__c = newCaseObj.ParentContentDocumentId__c;
}

if (savedScreen.size() > 0)
  newCaseObj.Saved_Screen__c = String.join(
    new List<String>(savedScreen),
    ','
  );

upsert newCaseObj;

upsert newApplicationParticipantList;

if (
  formCompleted != null &&
  formCompleted != '' &&
  String.isNotEmpty(formCompleted) &&
  String.isNotBlank(formCompleted) &&
  formCompleted == '100'
) {
  String emailId = newCaseObj.R_Email__c;
  SendEmailClass.sendEmailMethod1(
    'Application Form Completed',
    caseList[0],
    emailId
  );
  if (
    (socialWorkerContactObj != null &&
    socialWorkerContactObj != contactObj &&
    socialWorkerContactObj.Id != null &&
    socialWorkerContactObj.Email != null &&
    socialWorkerContactObj.Email != '' &&
    String.isNotEmpty(socialWorkerContactObj.Email) &&
    String.isNotBlank(socialWorkerContactObj.Email)) ||
    (doctorContactObj != null &&
```

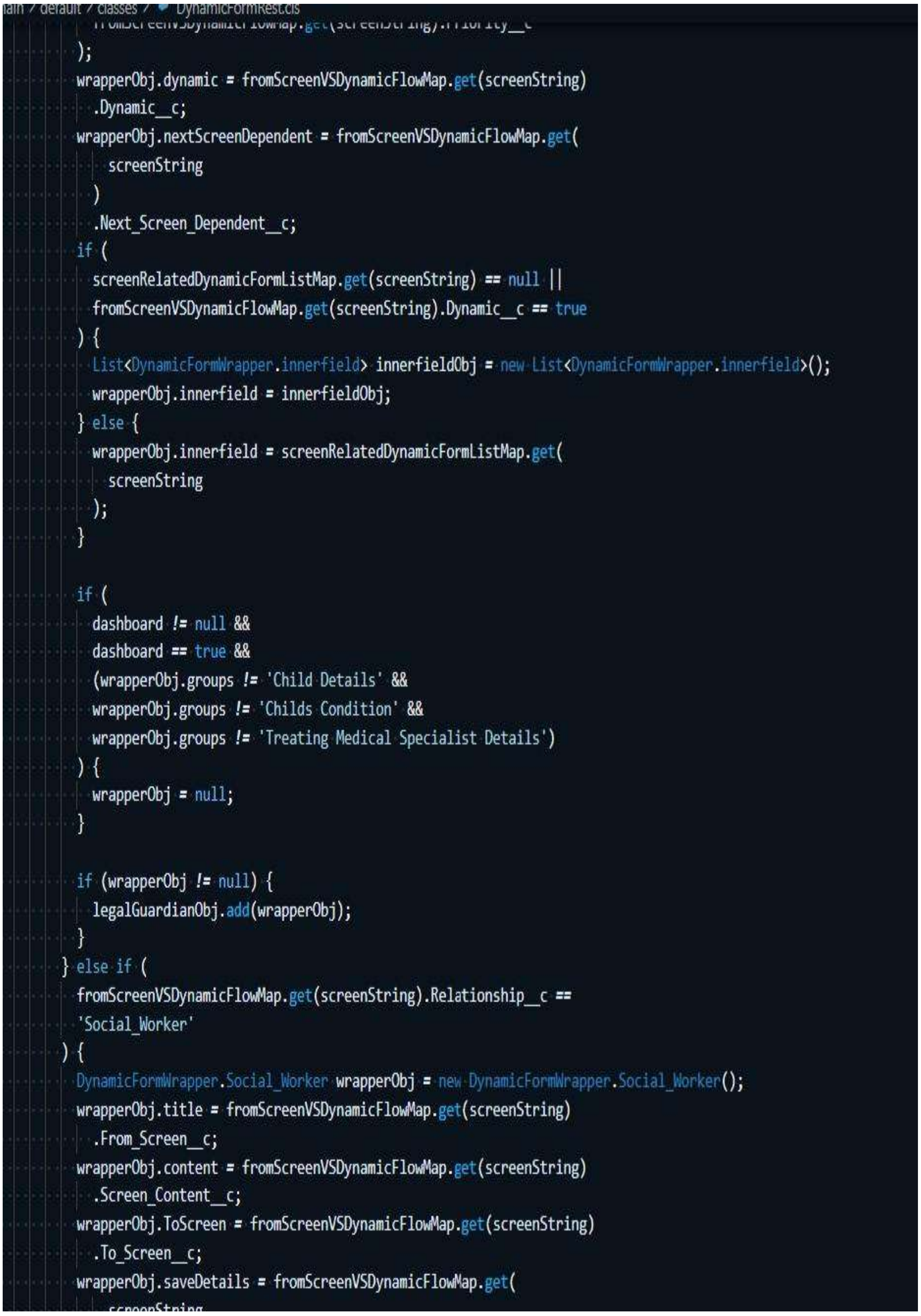


```
main > default > classes > DynamicFormRest.cls
    List<DynamicFormWrapper.Mother> motherObj = new List<DynamicFormWrapper.Mother>();
    List<DynamicFormWrapper.Father> fatherObj = new List<DynamicFormWrapper.Father>();
    List<DynamicFormWrapper.Legal_Guardian> legalGuardianObj = new List<DynamicFormWrapper.Legal_Guardian>();
    List<DynamicFormWrapper.Social_Worker> socialWorkerrObj = new List<DynamicFormWrapper.Social_Worker>();
    List<DynamicFormWrapper.Case_Worker> caseWorkerObj = new List<DynamicFormWrapper.Case_Worker>();
    List<DynamicFormWrapper.Doctor> DoctorObj = new List<DynamicFormWrapper.Doctor>();
    List<DynamicFormWrapper.Child> ChildObj = new List<DynamicFormWrapper.Child>();
    List<DynamicFormWrapper.Family_Members> familyMembersObj = new List<DynamicFormWrapper.Family_Members>();
    DynamicFormWrapper dynamicFormWrapperObj = new DynamicFormWrapper();
    for (String screenString : fromScreenVSDynamicFlowMap.keySet()) {
      if (fromScreenVSDynamicFlowMap.containsKey(screenString)) {
        if (fromScreenVSDynamicFlowMap.get(screenString) != null) {
          if (
            fromScreenVSDynamicFlowMap.get(screenString).Relationship__c ==
            'Mother'
          ) {
            DynamicFormWrapper.Mother wrapperObj = new DynamicFormWrapper.Mother();
            wrapperObj.title = fromScreenVSDynamicFlowMap.get(screenString)
              .From_Screen__c;
            wrapperObj.content = fromScreenVSDynamicFlowMap.get(screenString)
              .Screen_Content__c;
            wrapperObj.ToScreen = fromScreenVSDynamicFlowMap.get(screenString)
              .To_Screen__c;
            wrapperObj.saveDetails = fromScreenVSDynamicFlowMap.get(
                screenString
              )
              .Save_Details__c;
            if (
              savedScreens != null &&
              savedScreens.contains(
                fromScreenVSDynamicFlowMap.get(screenString).From_Screen__c
              )
            ) {
              wrapperObj.saved = true;
            } else {
              wrapperObj.saved = fromScreenVSDynamicFlowMap.get(screenString)
                .Saved__c;
            }

            wrapperObj.groups = fromScreenVSDynamicFlowMap.get(screenString)
              .Group__c;
            wrapperObj.priority = Integer.valueOf(
              fromScreenVSDynamicFlowMap.get(screenString).Priority__c
            );
            wrapperObj.dynamic = fromScreenVSDynamicFlowMap.get(screenString)
              .Dynamic__c;
```

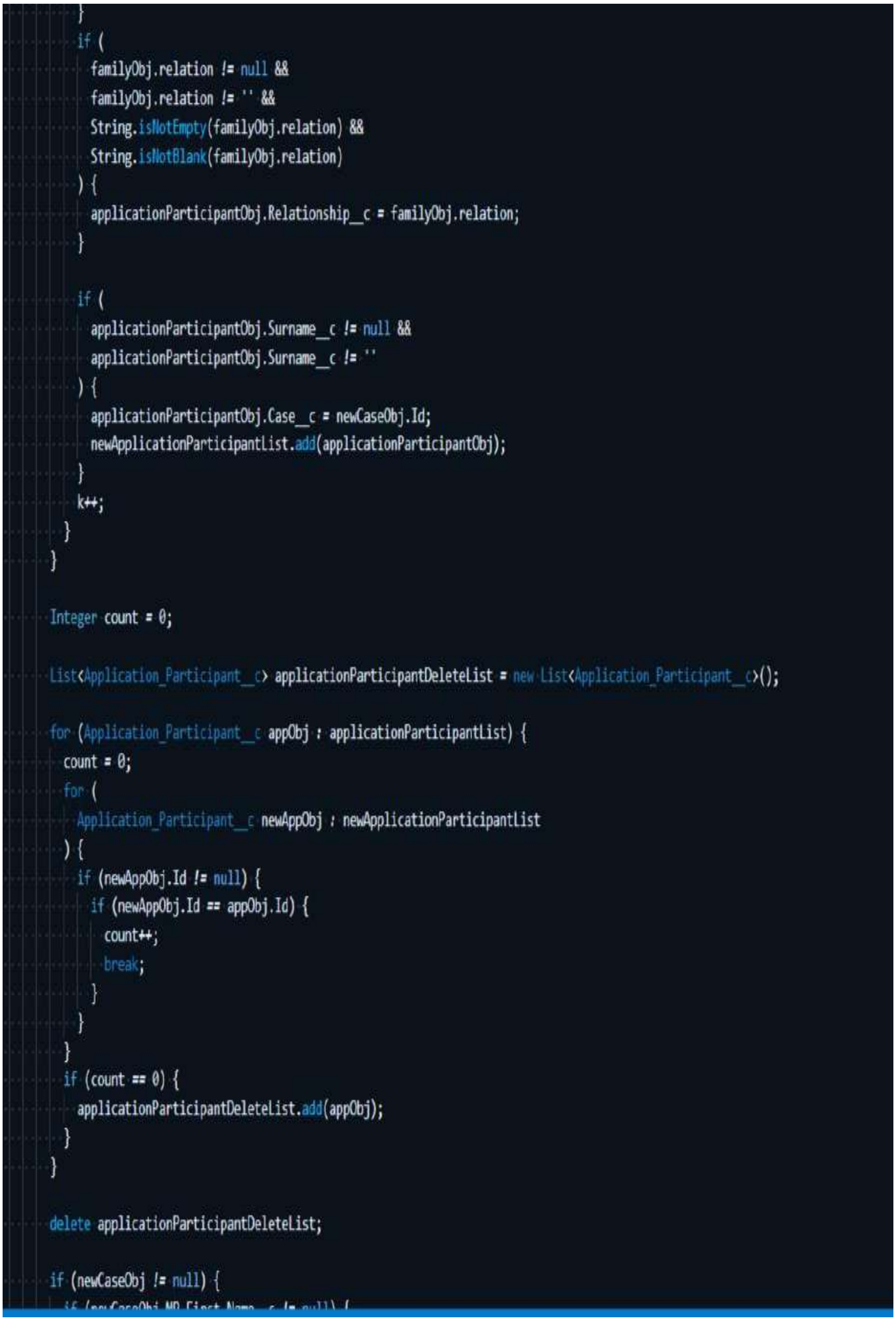


```
} else {
  caseQuery =
    'SELECT ' +
    caseFields +
    ', Specification_Related_Medical_Condition__c, Specification_Related_Wish__c, Fit_For_Wish__c, Child_Condition_by_Doctor__c FRO
    caseList[0].Id +
    '\' ORDER BY Id ASC LIMIT 1';
}

if (applicationParticipantList.size() > 0) {
  familyQuery =
    'SELECT ' +
    applicationParticipantFields +
    ' FROM Application_Participant__c WHERE Id IN: appParticipantIdSet LIMIT 5000';
}

caseResult = Database.query(caseQuery);
if (familyQuery != null) {
  familyResult = Database.query(familyQuery);
}

Integer i = 0;
for (Dynamic_Form__mdt dynamicFormObj : dynamicFormList) {
  if (
    !screenRelatedDynamicFormListMap.containsKey(
      dynamicFormObj.Relationship__c + dynamicFormObj.Screen__c
    )
  ) {
    screenRelatedDynamicFormListMap.put(
      dynamicFormObj.Relationship__c + dynamicFormObj.Screen__c,
      new List<DynamicFormWrapper.innerfield>()
    );
  }
  DynamicFormWrapper.innerfield innerfieldObj = new DynamicFormWrapper.innerfield();
  if (dynamicFormObj.Class__c == null) {
    innerfieldObj.cssClass = '';
  } else {
    innerfieldObj.cssClass = dynamicFormObj.Class__c;
  }
  innerfieldObj.disabled = dynamicFormObj.Disabled__c;
  innerfieldObj.field = dynamicFormObj.Field__c;
  if (dynamicFormObj.Label__c == null) {
    innerfieldObj.label = '';
  } else {
    innerfieldObj.label = dynamicFormObj.Label__c;
  }
```

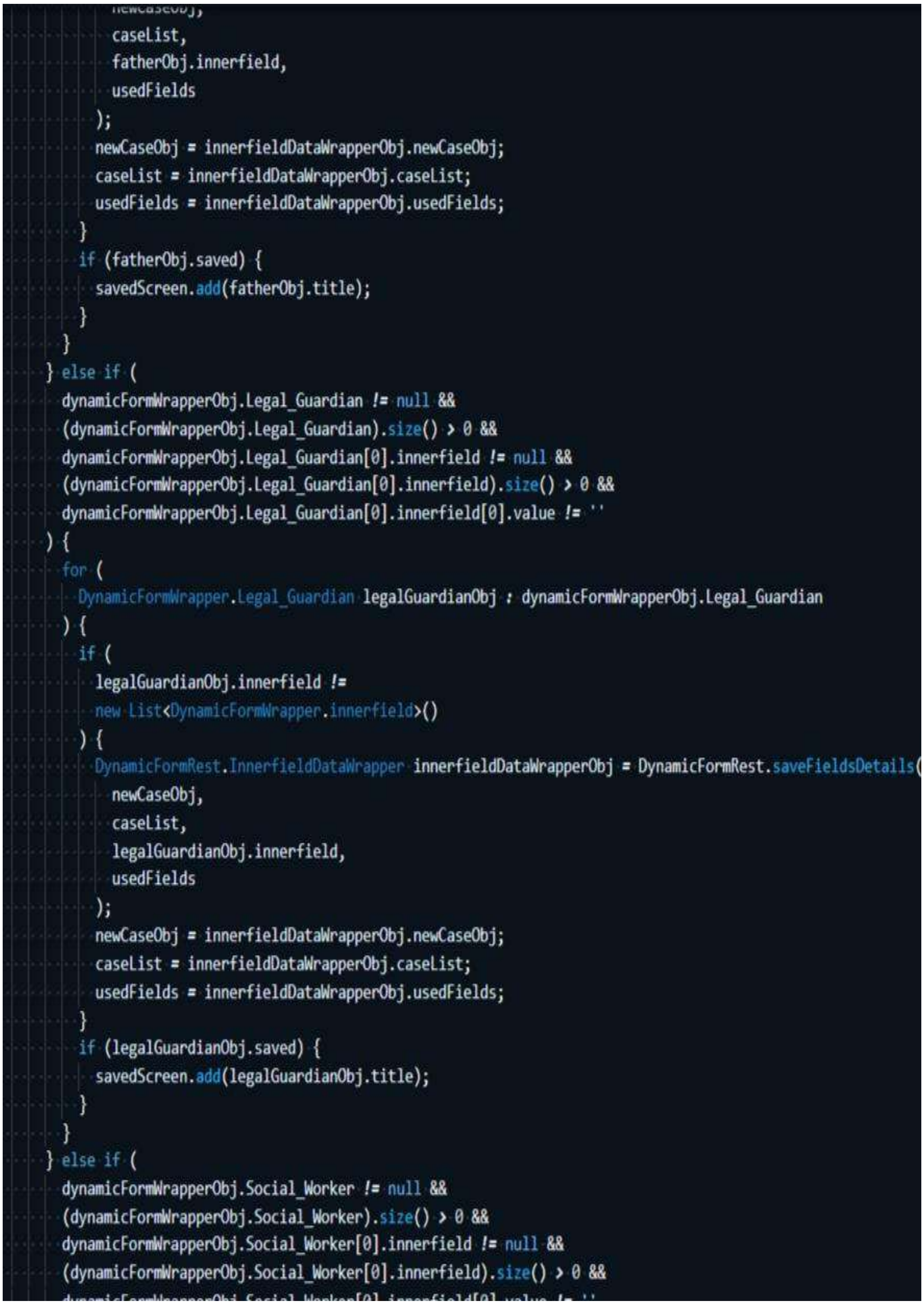

```
ist<DynamicFormWrapper.Mother> motherObj = new List<DynamicFormWrapper.Mother>();
ist<DynamicFormWrapper.Father> fatherObj = new List<DynamicFormWrapper.Father>();
ist<DynamicFormWrapper.Legal_Guardian> legalGuardianObj = new List<DynamicFormWrapper.Legal_Guardian>();
ist<DynamicFormWrapper.Social_Worker> socialWorkerrObj = new List<DynamicFormWrapper.Social_Worker>();
ist<DynamicFormWrapper.Case_Worker> caseWorkerObj = new List<DynamicFormWrapper.Case_Worker>();
ist<DynamicFormWrapper.Doctor> DoctorObj = new List<DynamicFormWrapper.Doctor>();
ist<DynamicFormWrapper.Child> ChildObj = new List<DynamicFormWrapper.Child>();
ist<DynamicFormWrapper.Family_Members> familyMembersObj = new List<DynamicFormWrapper.Family_Members>();
ynamicFormWrapper dynamicFormWrapperObj = new DynamicFormWrapper();
or (String screenString : fromScreenVSDynamicFlowMap.keySet()) {
  if (fromScreenVSDynamicFlowMap.containsKey(screenString)) {
    if (fromScreenVSDynamicFlowMap.get(screenString) != null) {
      if (
        fromScreenVSDynamicFlowMap.get(screenString).Relationship__c ==
        'Mother'
      ) {
        DynamicFormWrapper.Mother wrapperObj = new DynamicFormWrapper.Mother();
        wrapperObj.title = fromScreenVSDynamicFlowMap.get(screenString)
          .From_Screen__c;
        wrapperObj.content = fromScreenVSDynamicFlowMap.get(screenString)
          .Screen_Content__c;
        wrapperObj.ToScreen = fromScreenVSDynamicFlowMap.get(screenString)
          .To_Screen__c;
        wrapperObj.saveDetails = fromScreenVSDynamicFlowMap.get(
            screenString
          )
          .Save_Details__c;
        if (
          savedScreens != null &&
          savedScreens.contains(
            fromScreenVSDynamicFlowMap.get(screenString).From_Screen__c
          )
        ) {
          wrapperObj.saved = true;
        } else {
          wrapperObj.saved = fromScreenVSDynamicFlowMap.get(screenString)
            .Saved__c;
        }

        wrapperObj.groups = fromScreenVSDynamicFlowMap.get(screenString)
          .Group__c;
        wrapperObj.priority = Integer.valueOf(
          fromScreenVSDynamicFlowMap.get(screenString).Priority__c
        );
        wrapperObj.dynamic = fromScreenVSDynamicFlowMap.get(screenString)
          .Dynamic__c;
```



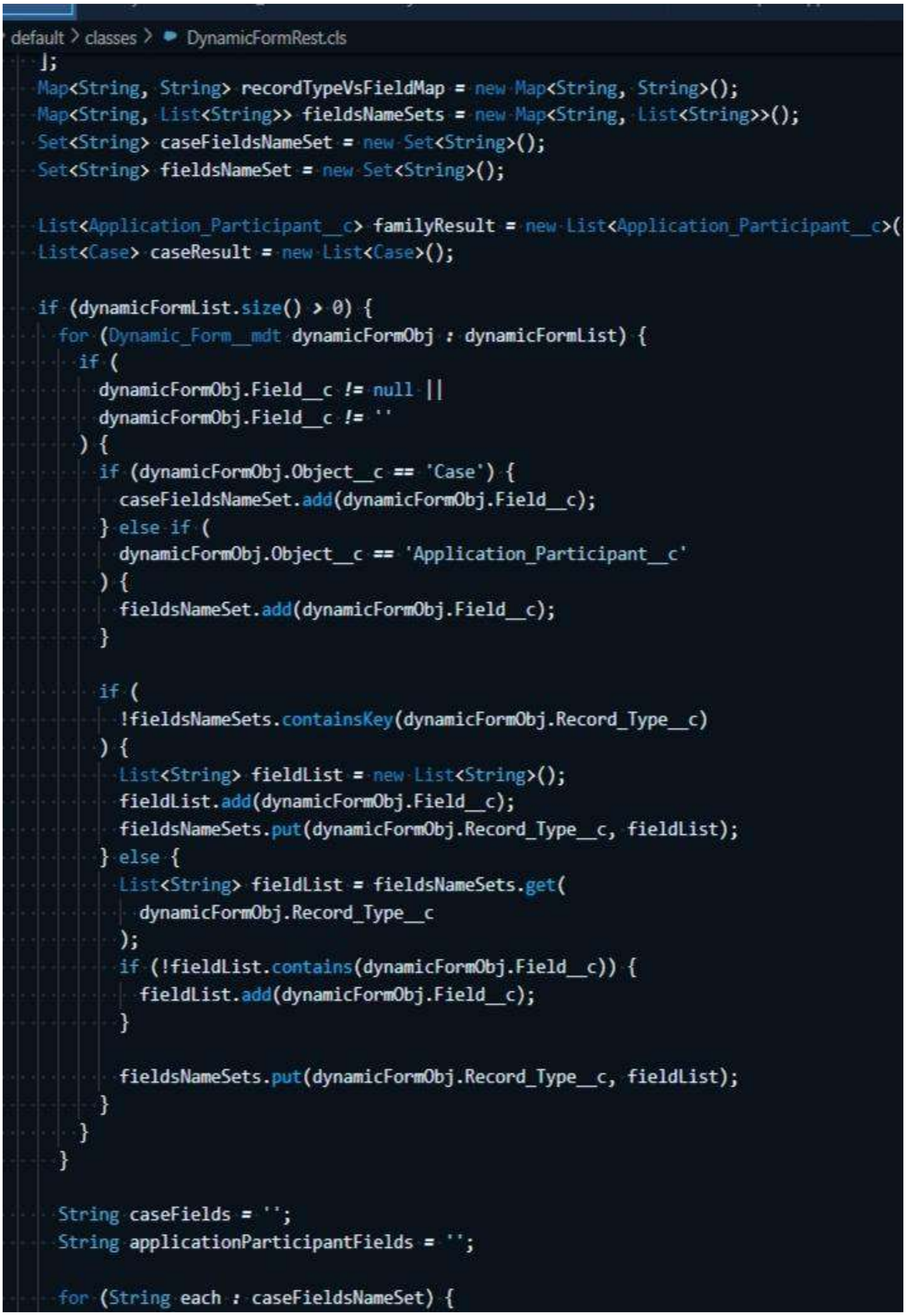
```
main / default / classes / DynamicFormRest.cls
      fromScreenVSDynamicFlowMap.get(screenString).Priority__c
    );
    wrapperObj.dynamic = fromScreenVSDynamicFlowMap.get(screenString)
      .Dynamic__c;
    wrapperObj.nextScreenDependent = fromScreenVSDynamicFlowMap.get(
        screenString
      )
      .Next_Screen_Dependent__c;
    if (
      screenRelatedDynamicFormListMap.get(screenString) == null ||
      fromScreenVSDynamicFlowMap.get(screenString).Dynamic__c == true
    ) {
      List<DynamicFormWrapper.innerfield> innerfieldObj = new List<DynamicFormWrapper.innerfield>();
      wrapperObj.innerfield = innerfieldObj;
    } else {
      wrapperObj.innerfield = screenRelatedDynamicFormListMap.get(
        screenString
      );
    }

    if (
      dashboard != null &&
      dashboard == true &&
      (wrapperObj.groups != 'Child Details' &&
      wrapperObj.groups != 'Childs Condition' &&
      wrapperObj.groups != 'Treating Medical Specialist Details')
    ) {
      wrapperObj = null;
    }

    if (wrapperObj != null) {
      legalGuardianObj.add(wrapperObj);
    }
  } else if (
    fromScreenVSDynamicFlowMap.get(screenString).Relationship__c ==
    'Social_Worker'
  ) {
    DynamicFormWrapper.Social_Worker wrapperObj = new DynamicFormWrapper.Social_Worker();
    wrapperObj.title = fromScreenVSDynamicFlowMap.get(screenString)
      .From_Screen__c;
    wrapperObj.content = fromScreenVSDynamicFlowMap.get(screenString)
      .Screen_Content__c;
    wrapperObj.ToScreen = fromScreenVSDynamicFlowMap.get(screenString)
      .To_Screen__c;
    wrapperObj.saveDetails = fromScreenVSDynamicFlowMap.get(
        screenString
```



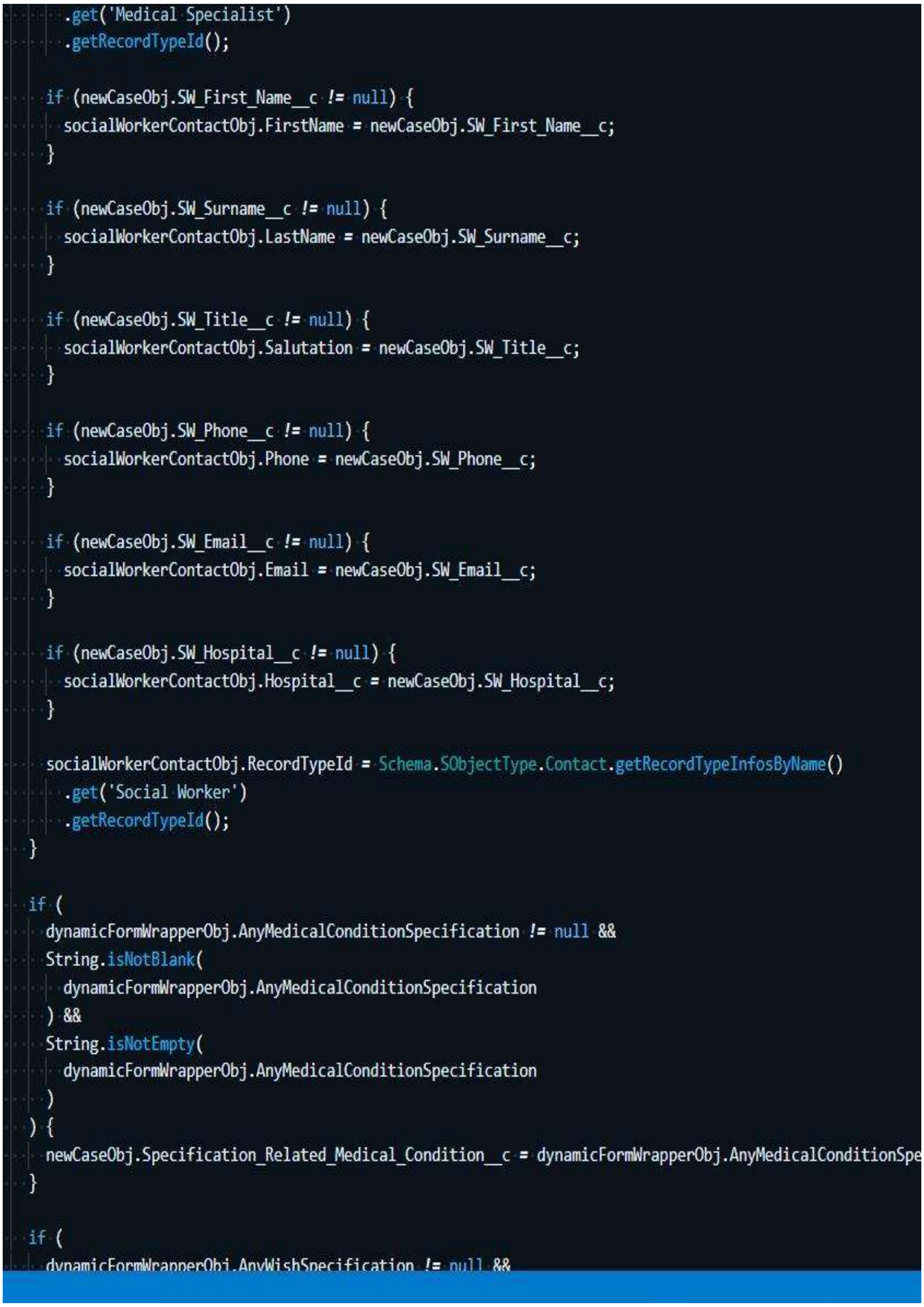
```
    }
    if (
      familyObj.relation != null &&
      familyObj.relation != '' &&
      String.isNotEmpty(familyObj.relation) &&
      String.isNotBlank(familyObj.relation)
    ) {
      applicationParticipantObj.Relationship__c = familyObj.relation;
    }

    if (
      applicationParticipantObj.Surname__c != null &&
      applicationParticipantObj.Surname__c != ''
    ) {
      applicationParticipantObj.Case__c = newCaseObj.Id;
      newApplicationParticipantList.add(applicationParticipantObj);
    }
    k++;
  }
}

Integer count = 0;

List<Application_Participant__c> applicationParticipantDeleteList = new List<Application_Participant__c>();

for (Application_Participant__c appObj : applicationParticipantList) {
  count = 0;
  for (
    Application_Participant__c newAppObj : newApplicationParticipantList
  ) {
    if (newAppObj.Id != null) {
      if (newAppObj.Id == appObj.Id) {
        count++;
        break;
      }
    }
  }
  if (count == 0) {
    applicationParticipantDeleteList.add(appObj);
  }
}

delete applicationParticipantDeleteList;

if (newCaseObj != null) {
  if (newCaseObj.MD_First_Name__c != null) {
```



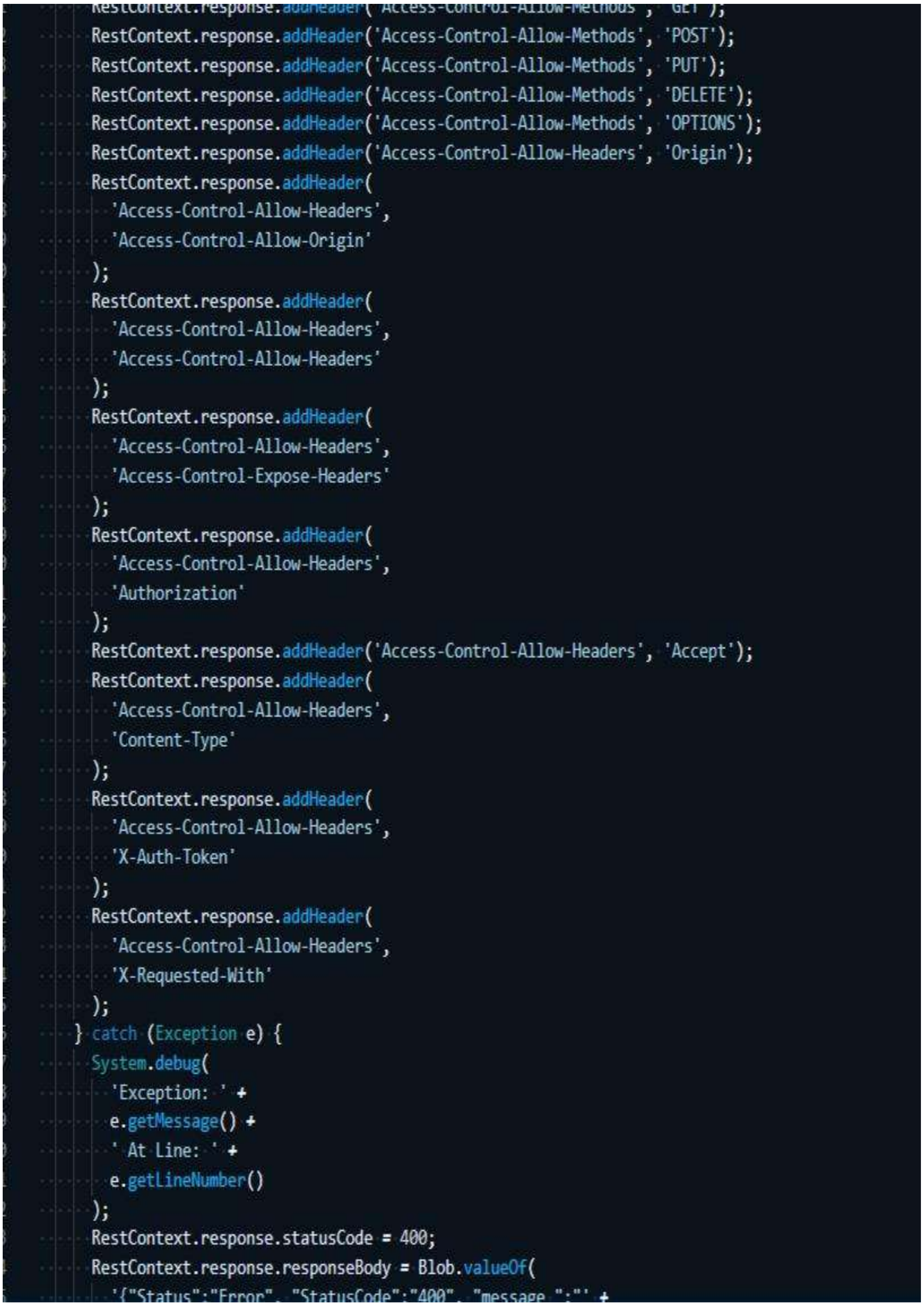
```
        newCaseObj,
        caseList,
        fatherObj.innerfield,
        usedFields
      );
      newCaseObj = innerfieldDataWrapperObj.newCaseObj;
      caseList = innerfieldDataWrapperObj.caseList;
      usedFields = innerfieldDataWrapperObj.usedFields;
    }
    if (fatherObj.saved) {
      savedScreen.add(fatherObj.title);
    }
  }
} else if (
  dynamicFormWrapperObj.Legal_Guardian != null &&
  (dynamicFormWrapperObj.Legal_Guardian).size() > 0 &&
  dynamicFormWrapperObj.Legal_Guardian[0].innerfield != null &&
  (dynamicFormWrapperObj.Legal_Guardian[0].innerfield).size() > 0 &&
  dynamicFormWrapperObj.Legal_Guardian[0].innerfield[0].value != ''
) {
  for (
    DynamicFormWrapper.Legal_Guardian legalGuardianObj : dynamicFormWrapperObj.Legal_Guardian
  ) {
    if (
      legalGuardianObj.innerfield !=
      new List<DynamicFormWrapper.innerfield>()
    ) {
      DynamicFormRest.InnerfieldDataWrapper innerfieldDataWrapperObj = DynamicFormRest.saveFieldsDetails(
        newCaseObj,
        caseList,
        legalGuardianObj.innerfield,
        usedFields
      );
      newCaseObj = innerfieldDataWrapperObj.newCaseObj;
      caseList = innerfieldDataWrapperObj.caseList;
      usedFields = innerfieldDataWrapperObj.usedFields;
    }
    if (legalGuardianObj.saved) {
      savedScreen.add(legalGuardianObj.title);
    }
  }
} else if (
  dynamicFormWrapperObj.Social_Worker != null &&
  (dynamicFormWrapperObj.Social_Worker).size() > 0 &&
  dynamicFormWrapperObj.Social_Worker[0].innerfield != null &&
  (dynamicFormWrapperObj.Social_Worker[0].innerfield).size() > 0 &&
  dynamicFormWrapperObj.Social_Worker[0].innerfield[0].value != ''
```


default > classes > DynamicFormRest.cls

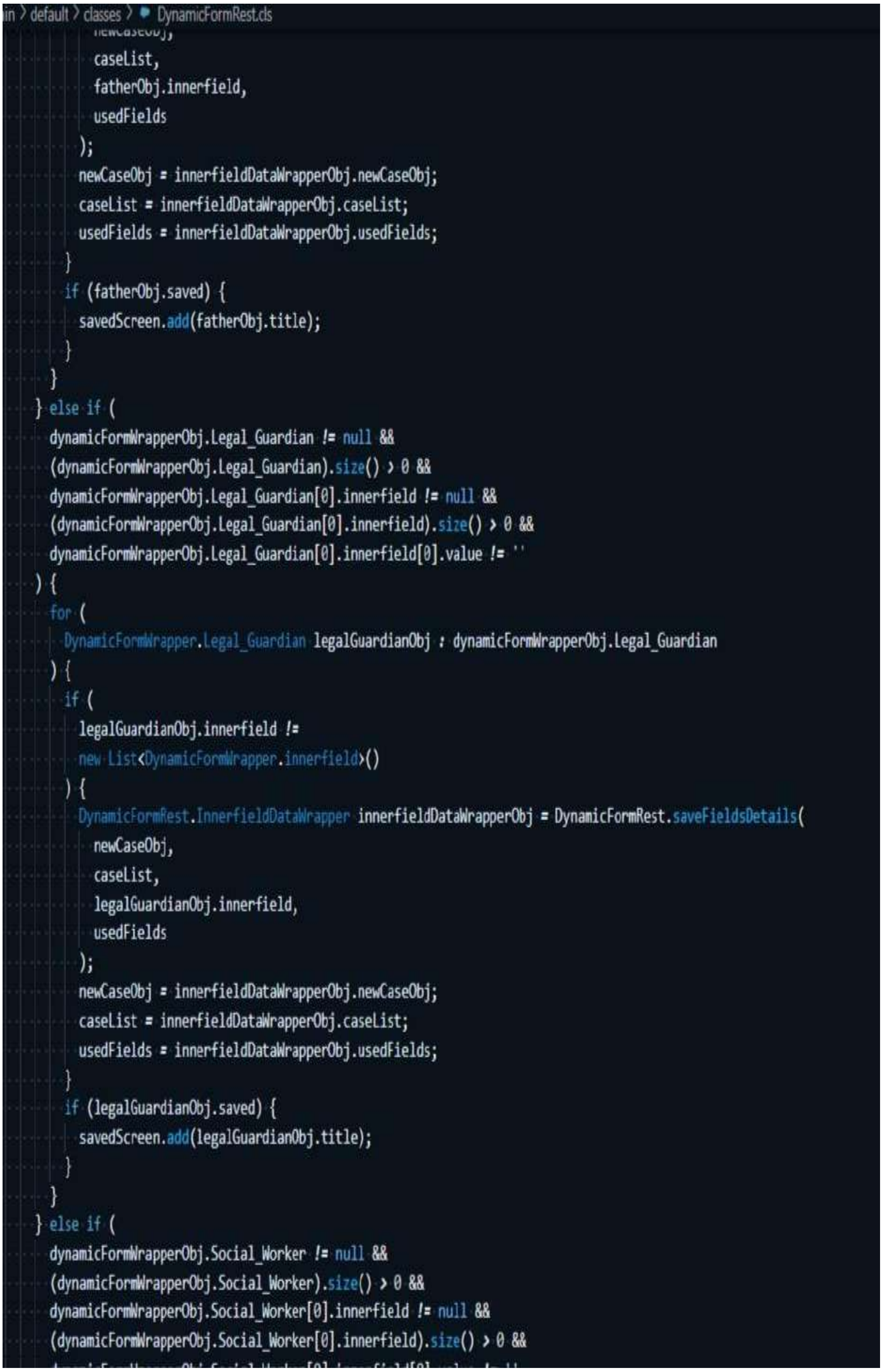
```
    ];
    Map<String, String> recordTypeVsFieldMap = new Map<String, String>();
    Map<String, List<String>> fieldsNameSets = new Map<String, List<String>>();
    Set<String> caseFieldsNameSet = new Set<String>();
    Set<String> fieldsNameSet = new Set<String>();

    List<Application_Participant__c> familyResult = new List<Application_Participant__c>(
    List<Case> caseResult = new List<Case>();

    if (dynamicFormList.size() > 0) {
      for (Dynamic_Form__mdt dynamicFormObj : dynamicFormList) {
        if (
          dynamicFormObj.Field__c != null ||
          dynamicFormObj.Field__c != ''
        ) {
          if (dynamicFormObj.Object__c == 'Case') {
            caseFieldsNameSet.add(dynamicFormObj.Field__c);
          } else if (
            dynamicFormObj.Object__c == 'Application_Participant__c'
          ) {
            fieldsNameSet.add(dynamicFormObj.Field__c);
          }

          if (
            !fieldsNameSets.containsKey(dynamicFormObj.Record_Type__c)
          ) {
            List<String> fieldList = new List<String>();
            fieldList.add(dynamicFormObj.Field__c);
            fieldsNameSets.put(dynamicFormObj.Record_Type__c, fieldList);
          } else {
            List<String> fieldList = fieldsNameSets.get(
              dynamicFormObj.Record_Type__c
            );
            if (!fieldList.contains(dynamicFormObj.Field__c)) {
              fieldList.add(dynamicFormObj.Field__c);
            }

            fieldsNameSets.put(dynamicFormObj.Record_Type__c, fieldList);
          }
        }
      }

      String caseFields = '';
      String applicationParticipantFields = '';

      for (String each : caseFieldsNameSet) {
```



```
            .get('Medical Specialist')
            .getRecordTypeId();

        if (newCaseObj.SW_First_Name__c != null) {
            socialWorkerContactObj.FirstName = newCaseObj.SW_First_Name__c;
        }

        if (newCaseObj.SW_Surname__c != null) {
            socialWorkerContactObj.LastName = newCaseObj.SW_Surname__c;
        }

        if (newCaseObj.SW_Title__c != null) {
            socialWorkerContactObj.Salutation = newCaseObj.SW_Title__c;
        }

        if (newCaseObj.SW_Phone__c != null) {
            socialWorkerContactObj.Phone = newCaseObj.SW_Phone__c;
        }

        if (newCaseObj.SW_Email__c != null) {
            socialWorkerContactObj.Email = newCaseObj.SW_Email__c;
        }

        if (newCaseObj.SW_Hospital__c != null) {
            socialWorkerContactObj.Hospital__c = newCaseObj.SW_Hospital__c;
        }

        socialWorkerContactObj.RecordTypeId = Schema.SObjectType.Contact.getRecordTypeInfosByName()
            .get('Social Worker')
            .getRecordTypeId();
    }

    if (
        dynamicFormWrapperObj.AnyMedicalConditionSpecification != null &&
        String.isNotBlank(
            dynamicFormWrapperObj.AnyMedicalConditionSpecification
        ) &&
        String.isNotEmpty(
            dynamicFormWrapperObj.AnyMedicalConditionSpecification
        )
    ) {
        newCaseObj.Specification_Related_Medical_Condition__c = dynamicFormWrapperObj.AnyMedicalConditionSpe
    }

    if (
        dynamicFormWrapperObj.AnyWishSpecification != null &&
```

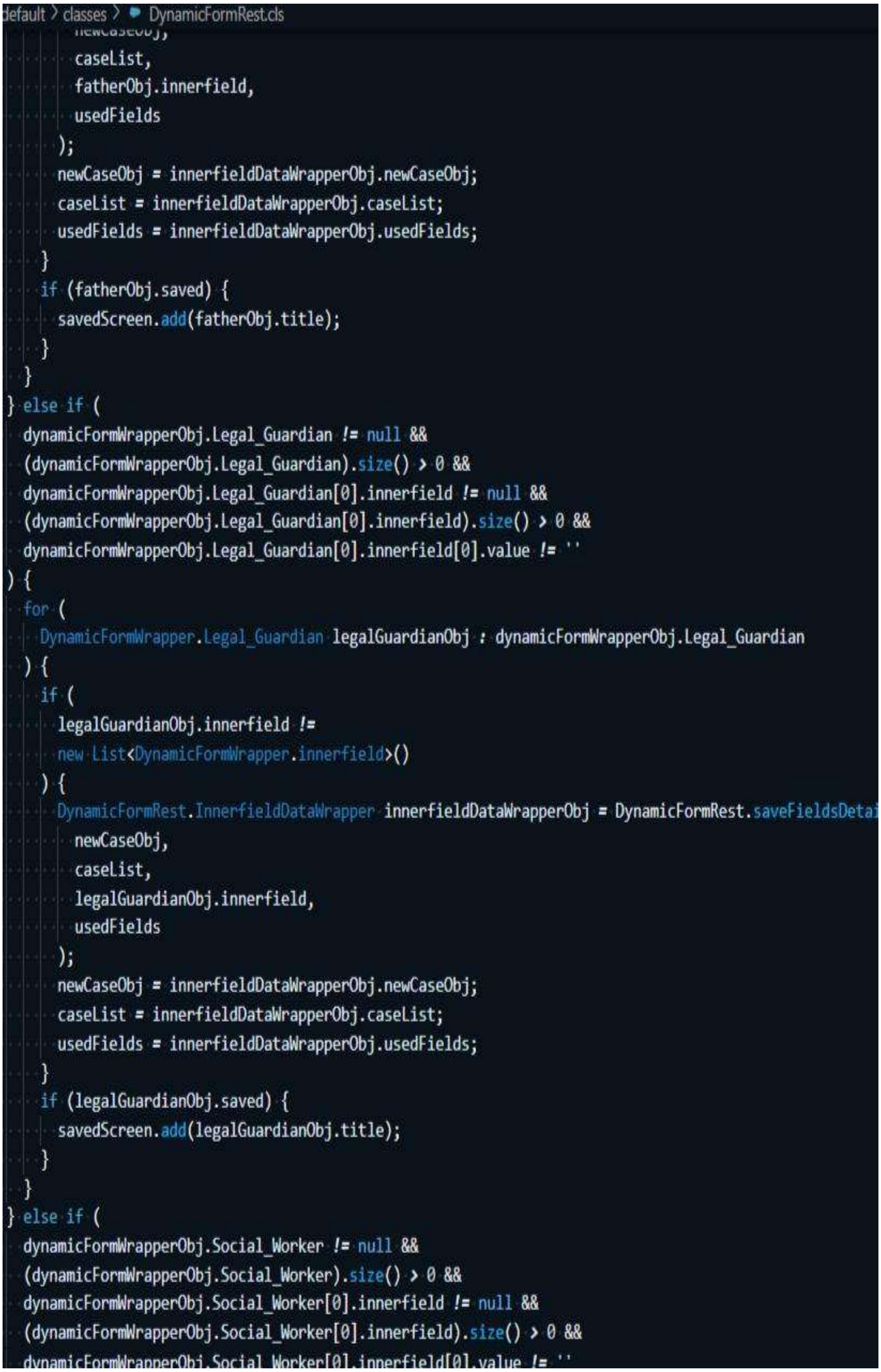


```
    RestContext.response.addHeader('Access-Control-Allow-Methods', 'GET');
    RestContext.response.addHeader('Access-Control-Allow-Methods', 'POST');
    RestContext.response.addHeader('Access-Control-Allow-Methods', 'PUT');
    RestContext.response.addHeader('Access-Control-Allow-Methods', 'DELETE');
    RestContext.response.addHeader('Access-Control-Allow-Methods', 'OPTIONS');
    RestContext.response.addHeader('Access-Control-Allow-Headers', 'Origin');
    RestContext.response.addHeader(
      'Access-Control-Allow-Headers',
      'Access-Control-Allow-Origin'
    );
    RestContext.response.addHeader(
      'Access-Control-Allow-Headers',
      'Access-Control-Allow-Headers'
    );
    RestContext.response.addHeader(
      'Access-Control-Allow-Headers',
      'Access-Control-Expose-Headers'
    );
    RestContext.response.addHeader(
      'Access-Control-Allow-Headers',
      'Authorization'
    );
    RestContext.response.addHeader('Access-Control-Allow-Headers', 'Accept');
    RestContext.response.addHeader(
      'Access-Control-Allow-Headers',
      'Content-Type'
    );
    RestContext.response.addHeader(
      'Access-Control-Allow-Headers',
      'X-Auth-Token'
    );
    RestContext.response.addHeader(
      'Access-Control-Allow-Headers',
      'X-Requested-With'
    );
  } catch (Exception e) {
    System.debug(
      'Exception: ' +
      e.getMessage() +
      ' At Line: ' +
      e.getLineNumber()
    );
    RestContext.response.statusCode = 400;
    RestContext.response.responseBody = Blob.valueOf(
      '{"Status":"Error", "StatusCode":"400", "message ":"' +
```

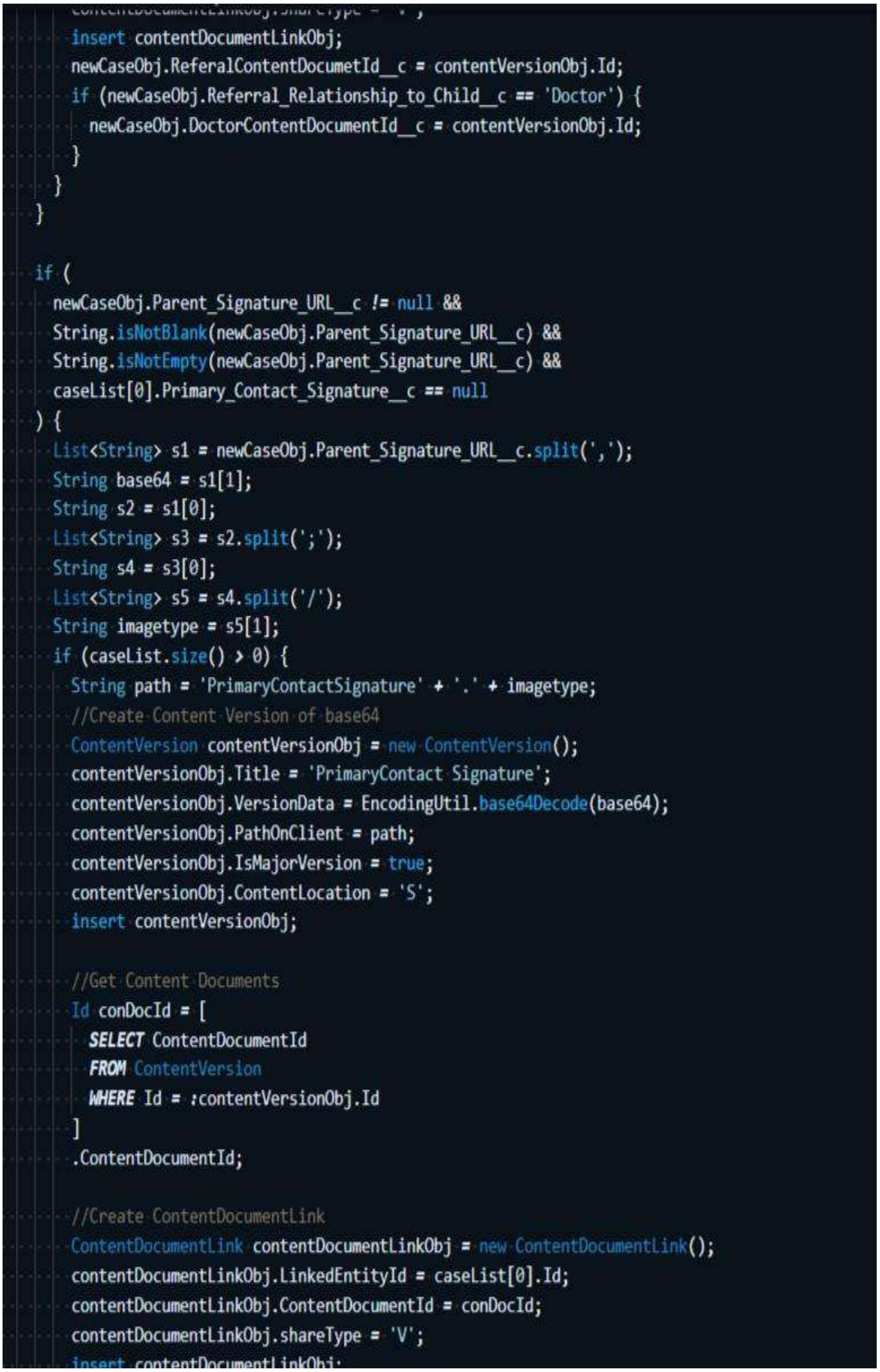


in > default > classes > DynamicFormRest.cls

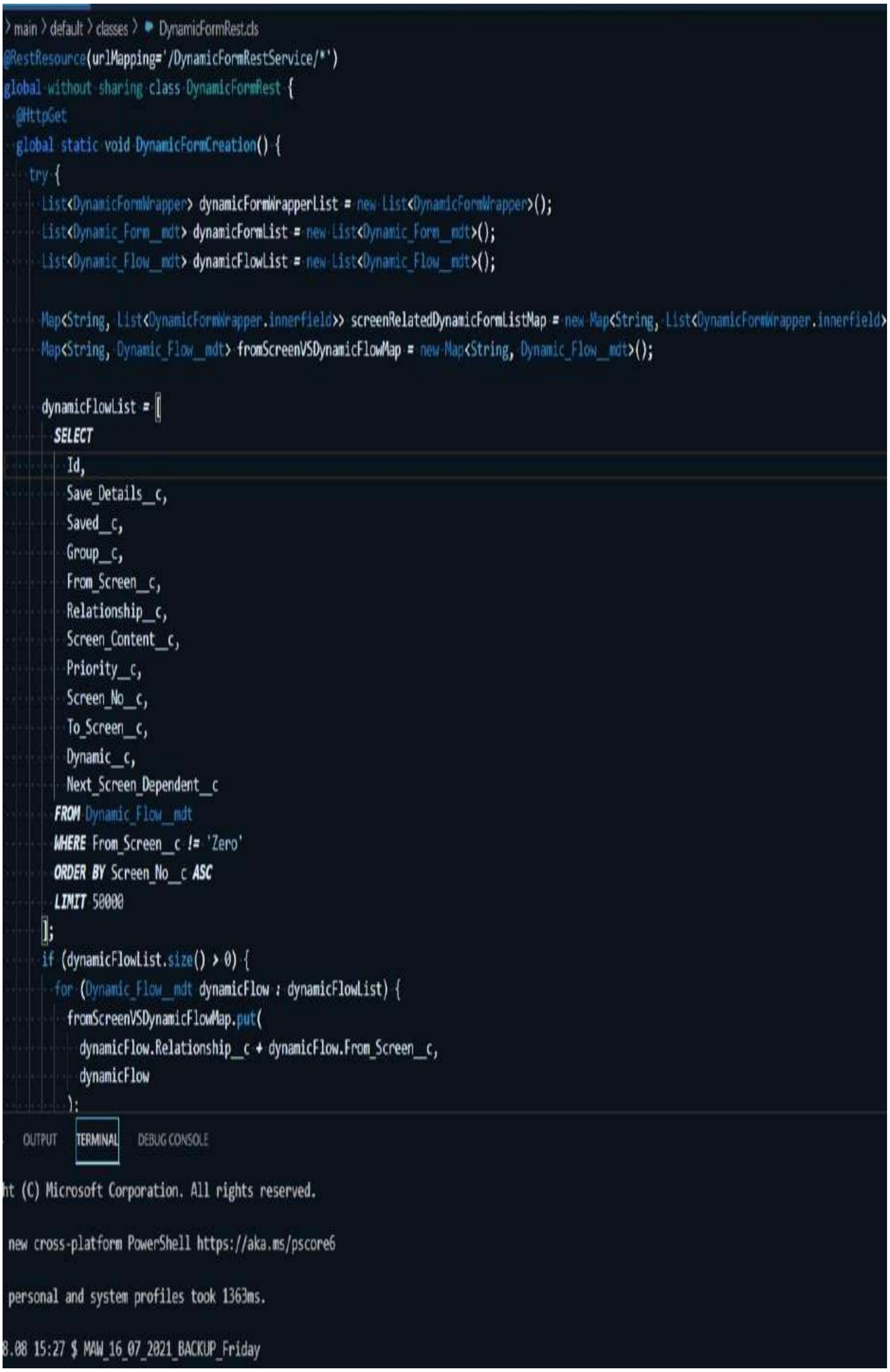
```
                    caseList,
                    fatherObj.innerfield,
                    usedFields
                );
                newCaseObj = innerfieldDataWrapperObj.newCaseObj;
                caseList = innerfieldDataWrapperObj.caseList;
                usedFields = innerfieldDataWrapperObj.usedFields;
            }
            if (fatherObj.saved) {
                savedScreen.add(fatherObj.title);
            }
        }
    } else if (
        dynamicFormWrapperObj.Legal_Guardian != null &&
        (dynamicFormWrapperObj.Legal_Guardian).size() > 0 &&
        dynamicFormWrapperObj.Legal_Guardian[0].innerfield != null &&
        (dynamicFormWrapperObj.Legal_Guardian[0].innerfield).size() > 0 &&
        dynamicFormWrapperObj.Legal_Guardian[0].innerfield[0].value != ''
    ) {
        for (
            DynamicFormWrapper.Legal_Guardian legalGuardianObj : dynamicFormWrapperObj.Legal_Guardian
        ) {
            if (
                legalGuardianObj.innerfield !=
                new List<DynamicFormWrapper.innerfield>()
            ) {
                DynamicFormRest.InnerfieldDataWrapper innerfieldDataWrapperObj = DynamicFormRest.saveFieldsDetails(
                    newCaseObj,
                    caseList,
                    legalGuardianObj.innerfield,
                    usedFields
                );
                newCaseObj = innerfieldDataWrapperObj.newCaseObj;
                caseList = innerfieldDataWrapperObj.caseList;
                usedFields = innerfieldDataWrapperObj.usedFields;
            }
            if (legalGuardianObj.saved) {
                savedScreen.add(legalGuardianObj.title);
            }
        }
    } else if (
        dynamicFormWrapperObj.Social_Worker != null &&
        (dynamicFormWrapperObj.Social_Worker).size() > 0 &&
        dynamicFormWrapperObj.Social_Worker[0].innerfield != null &&
        (dynamicFormWrapperObj.Social_Worker[0].innerfield).size() > 0 &&
```



```
default > classes > DynamicFormRest.cls
            caseList,
            fatherObj.innerfield,
            usedFields
          );
          newCaseObj = innerfieldDataWrapperObj.newCaseObj;
          caseList = innerfieldDataWrapperObj.caseList;
          usedFields = innerfieldDataWrapperObj.usedFields;
        }
        if (fatherObj.saved) {
          savedScreen.add(fatherObj.title);
        }
      }
} else if (
  dynamicFormWrapperObj.Legal_Guardian != null &&
  (dynamicFormWrapperObj.Legal_Guardian).size() > 0 &&
  dynamicFormWrapperObj.Legal_Guardian[0].innerfield != null &&
  (dynamicFormWrapperObj.Legal_Guardian[0].innerfield).size() > 0 &&
  dynamicFormWrapperObj.Legal_Guardian[0].innerfield[0].value != ''
) {
  for (
    DynamicFormWrapper.Legal_Guardian legalGuardianObj : dynamicFormWrapperObj.Legal_Guardian
  ) {
    if (
      legalGuardianObj.innerfield !=
      new List<DynamicFormWrapper.innerfield>()
    ) {
      DynamicFormRest.InnerfieldDataWrapper innerfieldDataWrapperObj = DynamicFormRest.saveFieldsDetai
        newCaseObj,
        caseList,
        legalGuardianObj.innerfield,
        usedFields
      );
      newCaseObj = innerfieldDataWrapperObj.newCaseObj;
      caseList = innerfieldDataWrapperObj.caseList;
      usedFields = innerfieldDataWrapperObj.usedFields;
    }
    if (legalGuardianObj.saved) {
      savedScreen.add(legalGuardianObj.title);
    }
  }
} else if (
  dynamicFormWrapperObj.Social_Worker != null &&
  (dynamicFormWrapperObj.Social_Worker).size() > 0 &&
  dynamicFormWrapperObj.Social_Worker[0].innerfield != null &&
  (dynamicFormWrapperObj.Social_Worker[0].innerfield).size() > 0 &&
  dynamicFormWrapperObj.Social_Worker[0].innerfield[0].value != ''
```

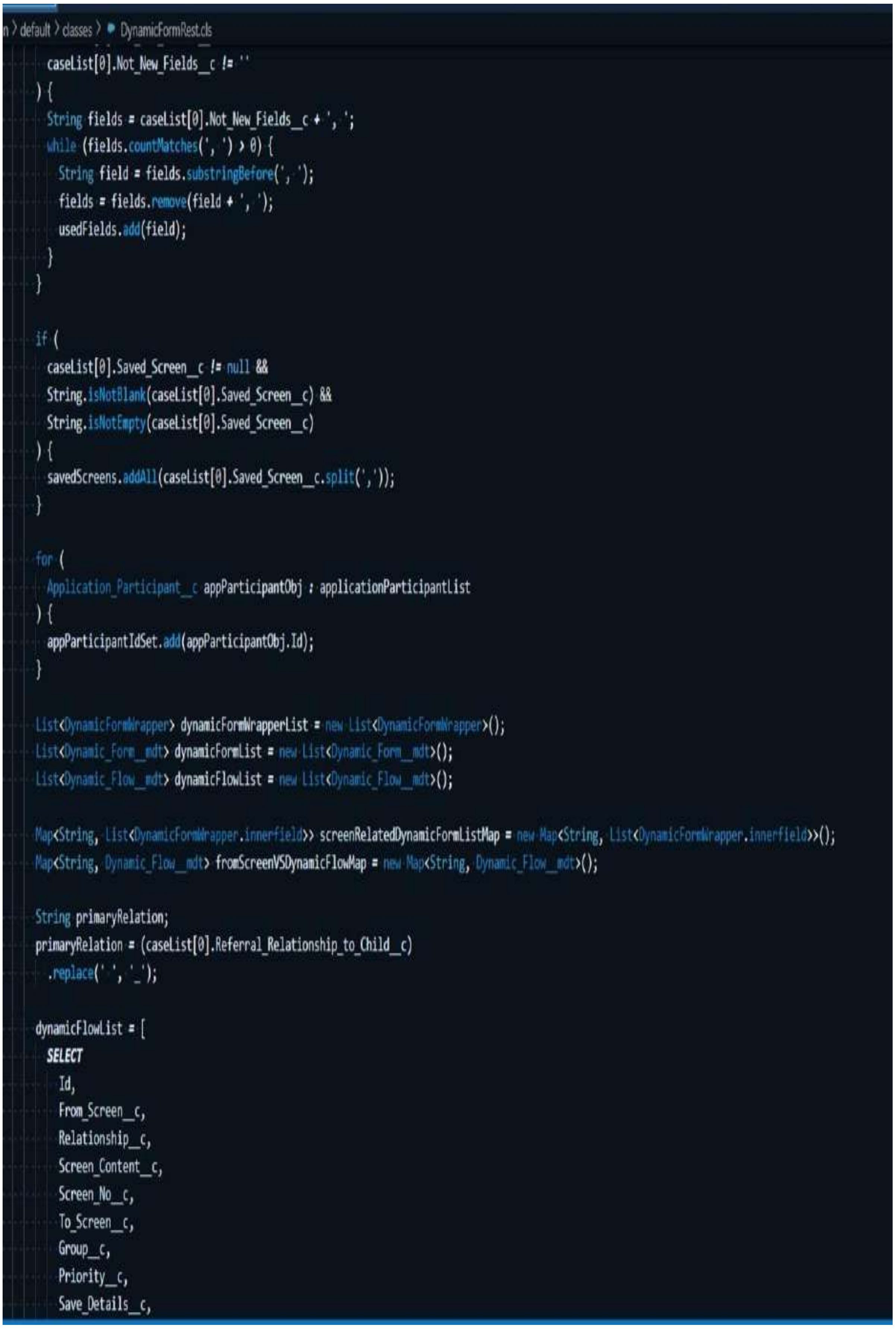


```
            contentDocumentLinkObj.shareType = 'V';
            insert contentDocumentLinkObj;
            newCaseObj.ReferalContentDocumetId__c = contentVersionObj.Id;
            if (newCaseObj.Referral_Relationship_to_Child__c == 'Doctor') {
                newCaseObj.DoctorContentDocumentId__c = contentVersionObj.Id;
            }
        }
    }

    if (
        newCaseObj.Parent_Signature_URL__c != null &&
        String.isNotBlank(newCaseObj.Parent_Signature_URL__c) &&
        String.isNotEmpty(newCaseObj.Parent_Signature_URL__c) &&
        caseList[0].Primary_Contact_Signature__c == null
    ) {
        List<String> s1 = newCaseObj.Parent_Signature_URL__c.split(',');
        String base64 = s1[1];
        String s2 = s1[0];
        List<String> s3 = s2.split(';');
        String s4 = s3[0];
        List<String> s5 = s4.split('/');
        String imagetype = s5[1];
        if (caseList.size() > 0) {
            String path = 'PrimaryContactSignature' + '.' + imagetype;
            //Create Content Version of base64
            ContentVersion contentVersionObj = new ContentVersion();
            contentVersionObj.Title = 'PrimaryContact Signature';
            contentVersionObj.VersionData = EncodingUtil.base64Decode(base64);
            contentVersionObj.PathOnClient = path;
            contentVersionObj.IsMajorVersion = true;
            contentVersionObj.ContentLocation = 'S';
            insert contentVersionObj;

            //Get Content Documents
            Id conDocId = [
                SELECT ContentDocumentId
                FROM ContentVersion
                WHERE Id = :contentVersionObj.Id
            ]
            .ContentDocumentId;

            //Create ContentDocumentLink
            ContentDocumentLink contentDocumentLinkObj = new ContentDocumentLink();
            contentDocumentLinkObj.LinkedEntityId = caseList[0].Id;
            contentDocumentLinkObj.ContentDocumentId = conDocId;
            contentDocumentLinkObj.shareType = 'V';
            insert contentDocumentLinkObj;
```

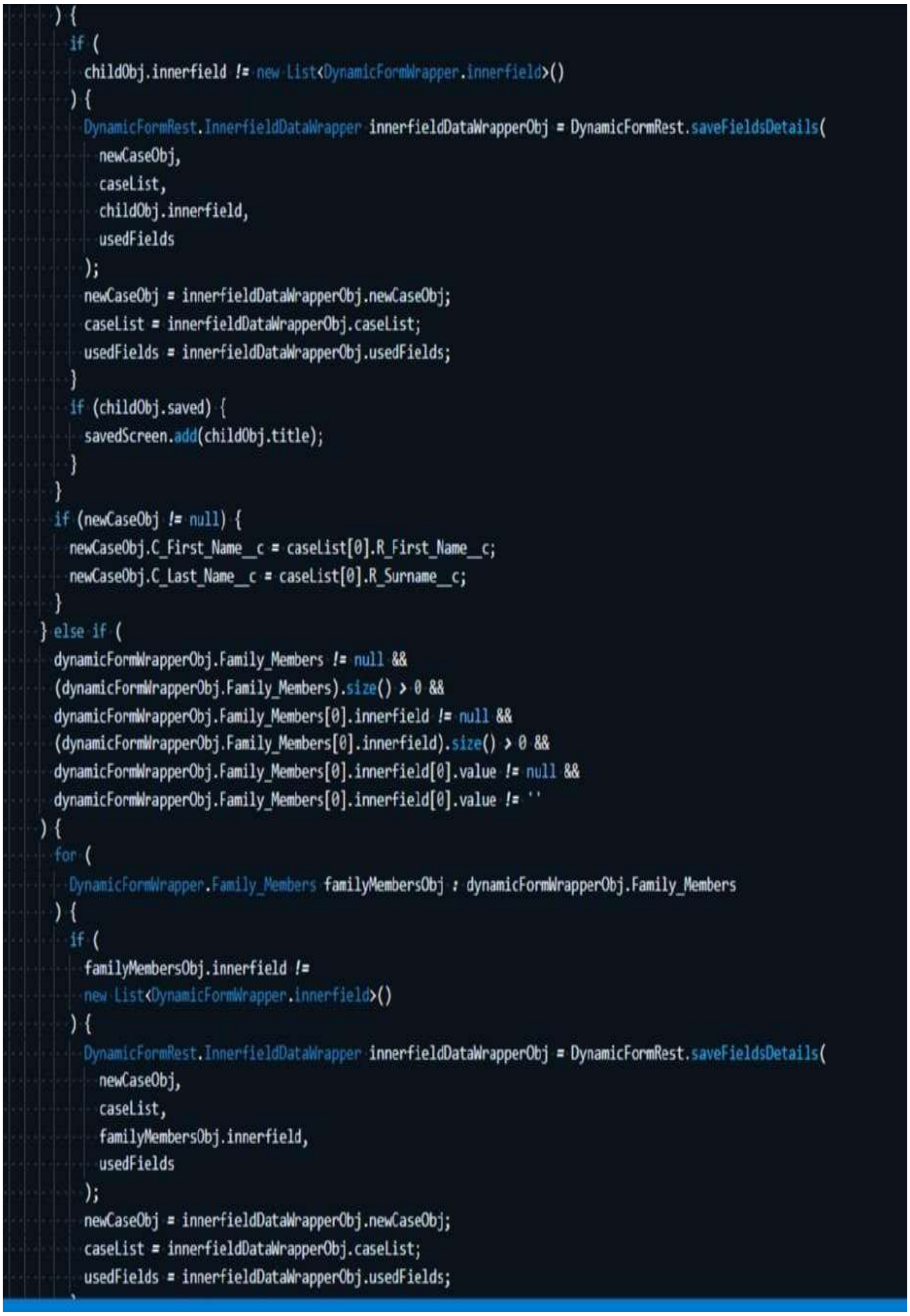


```
@RestResource(urlMapping='/DynamicFormRestService/*')
global without sharing class DynamicFormRest {
  @HttpGet
  global static void DynamicFormCreation() {
    try {
      List<DynamicFormWrapper> dynamicFormWrapperList = new List<DynamicFormWrapper>();
      List<Dynamic_Form__mdt> dynamicFormList = new List<Dynamic_Form__mdt>();
      List<Dynamic_Flow__mdt> dynamicFlowList = new List<Dynamic_Flow__mdt>();

      Map<String, List<DynamicFormWrapper.innerfield>> screenRelatedDynamicFormListMap = new Map<String, List<DynamicFormWrapper.innerfield>
      Map<String, Dynamic_Flow__mdt> fromScreenVSDynamicFlowMap = new Map<String, Dynamic_Flow__mdt>();

      dynamicFlowList = [
        SELECT
          Id,
          Save_Details__c,
          Saved__c,
          Group__c,
          From_Screen__c,
          Relationship__c,
          Screen_Content__c,
          Priority__c,
          Screen_No__c,
          To_Screen__c,
          Dynamic__c,
          Next_Screen_Dependent__c
        FROM Dynamic_Flow__mdt
        WHERE From_Screen__c != 'Zero'
        ORDER BY Screen_No__c ASC
        LIMIT 50000
      ];
      if (dynamicFlowList.size() > 0) {
        for (Dynamic_Flow__mdt dynamicFlow : dynamicFlowList) {
          fromScreenVSDynamicFlowMap.put(
            dynamicFlow.Relationship__c + dynamicFlow.From_Screen__c,
            dynamicFlow
          );
```

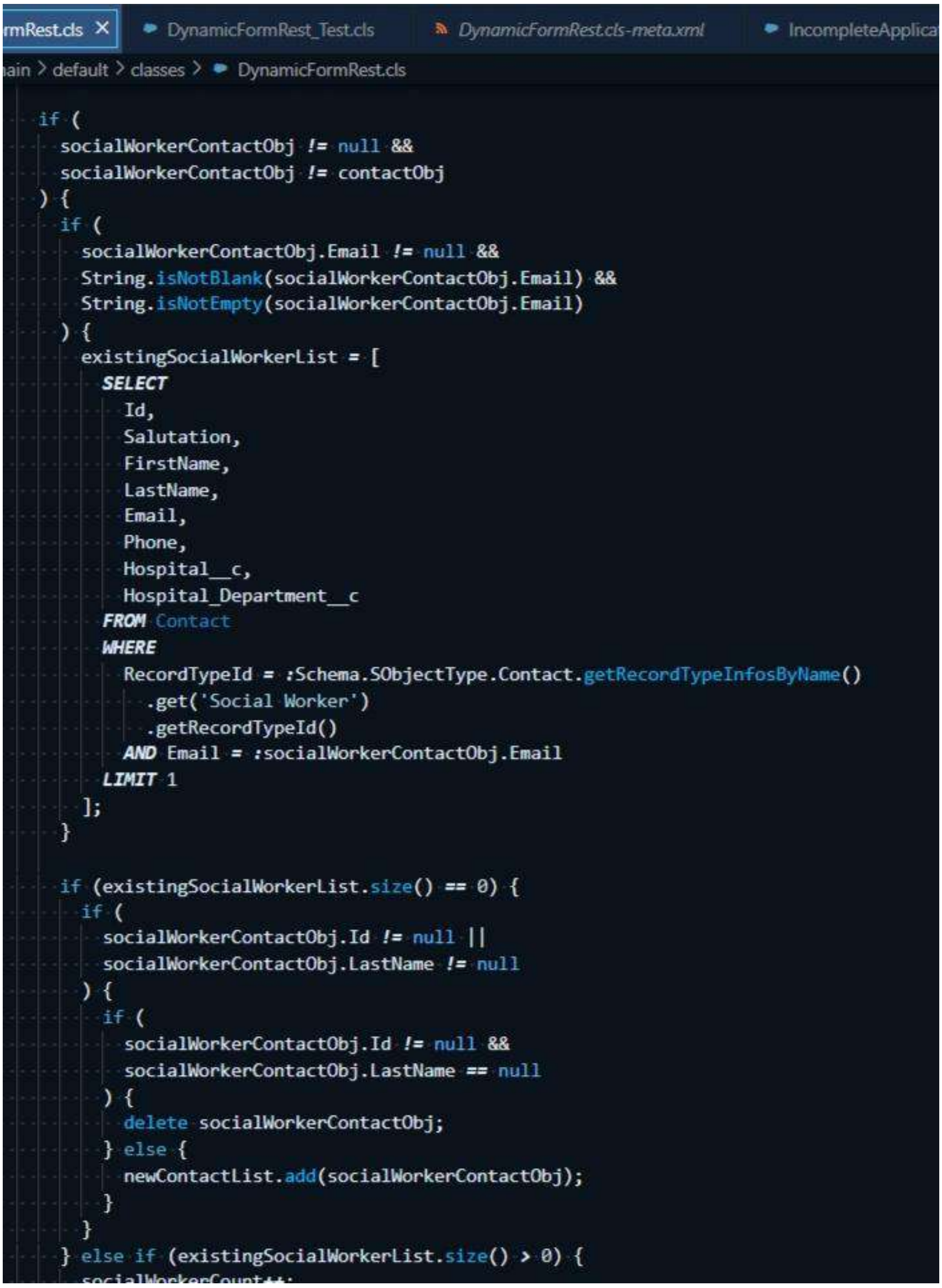
```
ht (C) Microsoft Corporation. All rights reserved.

 new cross-platform PowerShell https://aka.ms/pscore6

 personal and system profiles took 1363ms.

8.08 15:27 $ MAW_16_07_2021_BACKUP_Friday
```



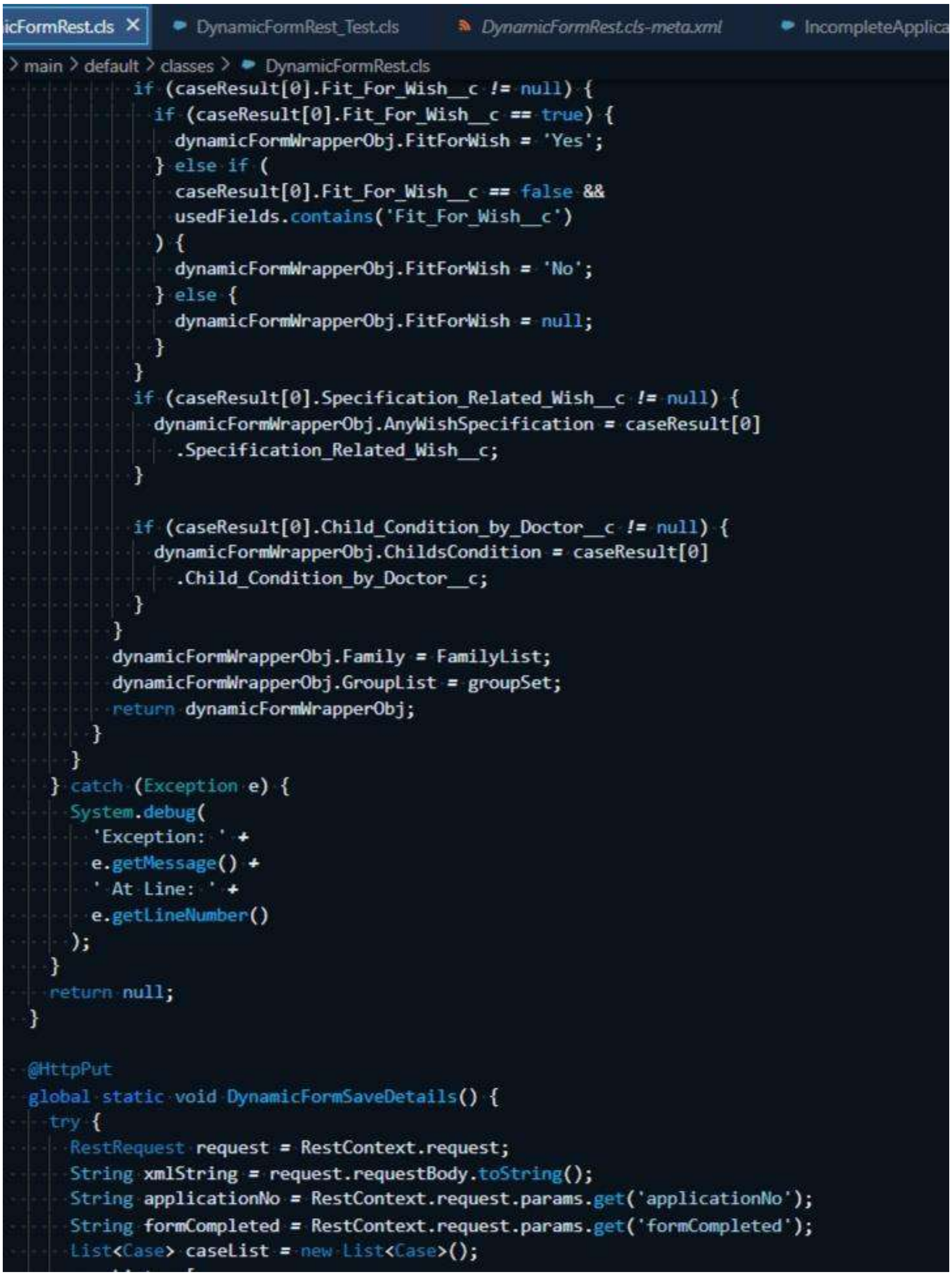
```
    caseList[0].Not_New_Fields__c != ''
  ) {
    String fields = caseList[0].Not_New_Fields__c + ', ';
    while (fields.countMatches(', ') > 0) {
      String field = fields.substringBefore(', ');
      fields = fields.remove(field + ', ');
      usedFields.add(field);
    }
  }

  if (
    caseList[0].Saved_Screen__c != null &&
    String.isNotBlank(caseList[0].Saved_Screen__c) &&
    String.isNotEmpty(caseList[0].Saved_Screen__c)
  ) {
    savedScreens.addAll(caseList[0].Saved_Screen__c.split(','));
  }

  for (
    Application_Participant__c appParticipantObj : applicationParticipantList
  ) {
    appParticipantIdSet.add(appParticipantObj.Id);
  }

  List<DynamicFormWrapper> dynamicFormWrapperList = new List<DynamicFormWrapper>();
  List<Dynamic_Form__mdt> dynamicFormList = new List<Dynamic_Form__mdt>();
  List<Dynamic_Flow__mdt> dynamicFlowList = new List<Dynamic_Flow__mdt>();

  Map<String, List<DynamicFormWrapper.innerfield>> screenRelatedDynamicFormListMap = new Map<String, List<DynamicFormWrapper.innerfield>>();
  Map<String, Dynamic_Flow__mdt> fromScreenVSDynamicFlowMap = new Map<String, Dynamic_Flow__mdt>();

  String primaryRelation;
  primaryRelation = (caseList[0].Referral_Relationship_to_Child__c)
    .replace(' ', '_');

  dynamicFlowList = [
    SELECT
      Id,
      From_Screen__c,
      Relationship__c,
      Screen_Content__c,
      Screen_No__c,
      To_Screen__c,
      Group__c,
      Priority__c,
      Save_Details__c,
```



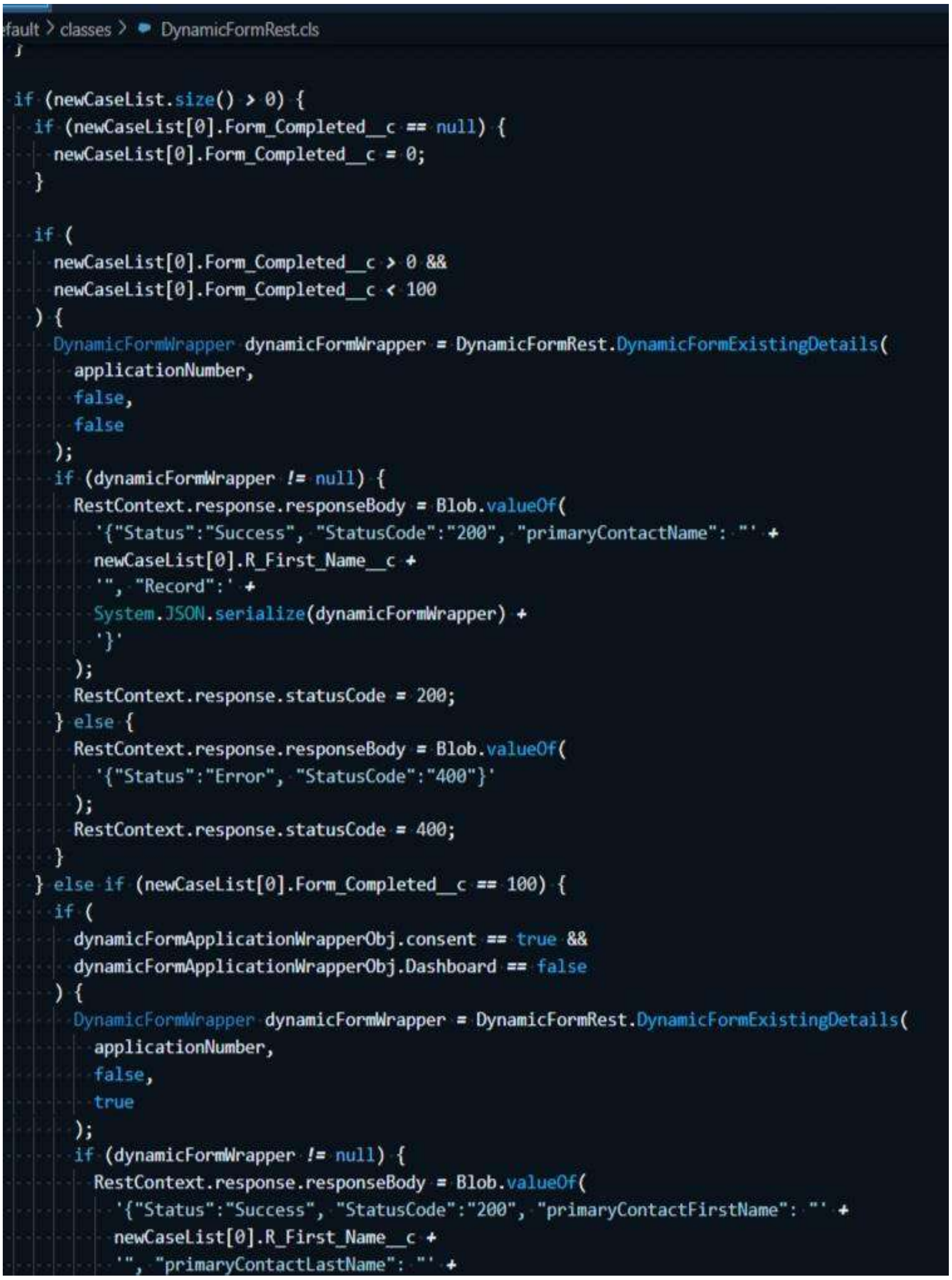
```
        ) {
            if (
                childObj.innerfield != new List<DynamicFormWrapper.innerfield>()
            ) {
                DynamicFormRest.InnerfieldDataWrapper innerfieldDataWrapperObj = DynamicFormRest.saveFieldsDetails(
                    newCaseObj,
                    caseList,
                    childObj.innerfield,
                    usedFields
                );
                newCaseObj = innerfieldDataWrapperObj.newCaseObj;
                caseList = innerfieldDataWrapperObj.caseList;
                usedFields = innerfieldDataWrapperObj.usedFields;
            }
            if (childObj.saved) {
                savedScreen.add(childObj.title);
            }
        }
        if (newCaseObj != null) {
            newCaseObj.C_First_Name__c = caseList[0].R_First_Name__c;
            newCaseObj.C_Last_Name__c = caseList[0].R_Surname__c;
        }
    } else if (
        dynamicFormWrapperObj.Family_Members != null &&
        (dynamicFormWrapperObj.Family_Members).size() > 0 &&
        dynamicFormWrapperObj.Family_Members[0].innerfield != null &&
        (dynamicFormWrapperObj.Family_Members[0].innerfield).size() > 0 &&
        dynamicFormWrapperObj.Family_Members[0].innerfield[0].value != null &&
        dynamicFormWrapperObj.Family_Members[0].innerfield[0].value != ''
    ) {
        for (
            DynamicFormWrapper.Family_Members familyMembersObj : dynamicFormWrapperObj.Family_Members
        ) {
            if (
                familyMembersObj.innerfield !=
                new List<DynamicFormWrapper.innerfield>()
            ) {
                DynamicFormRest.InnerfieldDataWrapper innerfieldDataWrapperObj = DynamicFormRest.saveFieldsDetails(
                    newCaseObj,
                    caseList,
                    familyMembersObj.innerfield,
                    usedFields
                );
                newCaseObj = innerfieldDataWrapperObj.newCaseObj;
                caseList = innerfieldDataWrapperObj.caseList;
                usedFields = innerfieldDataWrapperObj.usedFields;
```


rmRest.cls ×
DynamicFormRest_Test.cls
DynamicFormRest.cls-meta.xml
IncompleteApplica
ain > default > classes > DynamicFormRest.cls

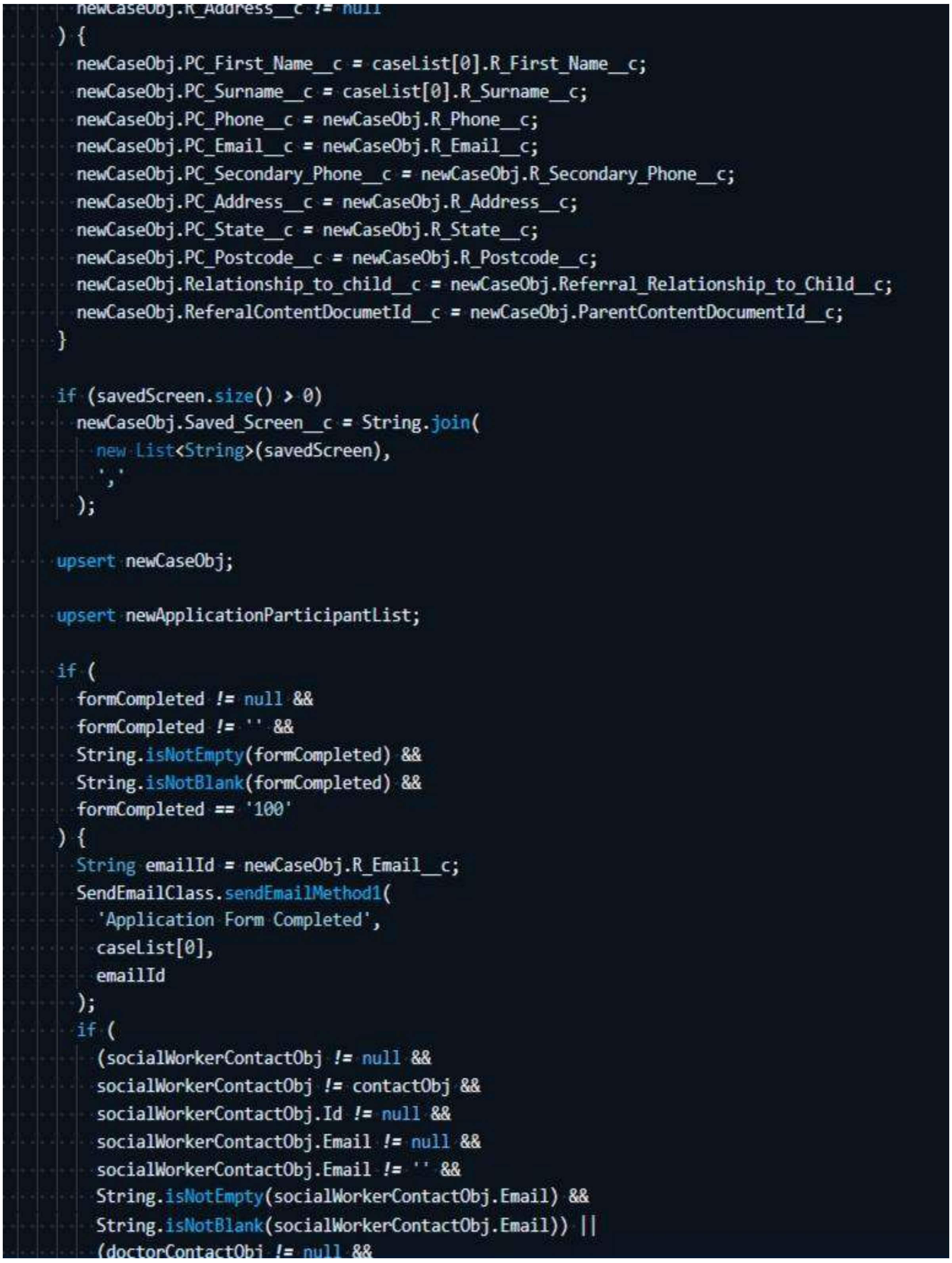
```
  if (
    socialWorkerContactObj != null &&
    socialWorkerContactObj != contactObj
  ) {
    if (
      socialWorkerContactObj.Email != null &&
      String.isNotBlank(socialWorkerContactObj.Email) &&
      String.isNotEmpty(socialWorkerContactObj.Email)
    ) {
      existingSocialWorkerList = [
        SELECT
          Id,
          Salutation,
          FirstName,
          LastName,
          Email,
          Phone,
          Hospital__c,
          Hospital_Department__c
        FROM Contact
        WHERE
          RecordTypeId = :Schema.SObjectType.Contact.getRecordTypeInfosByName()
            .get('Social Worker')
            .getRecordTypeId()
          AND Email = :socialWorkerContactObj.Email
        LIMIT 1
      ];
    }

    if (existingSocialWorkerList.size() == 0) {
      if (
        socialWorkerContactObj.Id != null ||
        socialWorkerContactObj.LastName != null
      ) {
        if (
          socialWorkerContactObj.Id != null &&
          socialWorkerContactObj.LastName == null
        ) {
          delete socialWorkerContactObj;
        } else {
          newContactList.add(socialWorkerContactObj);
        }
      }
    } else if (existingSocialWorkerList.size() > 0) {
```


icFormRest.cls
DynamicFormRest_Test.cls
DynamicFormRest.cls-meta.xml
IncompleteApplica
main > default > classes > DynamicFormRest.cls

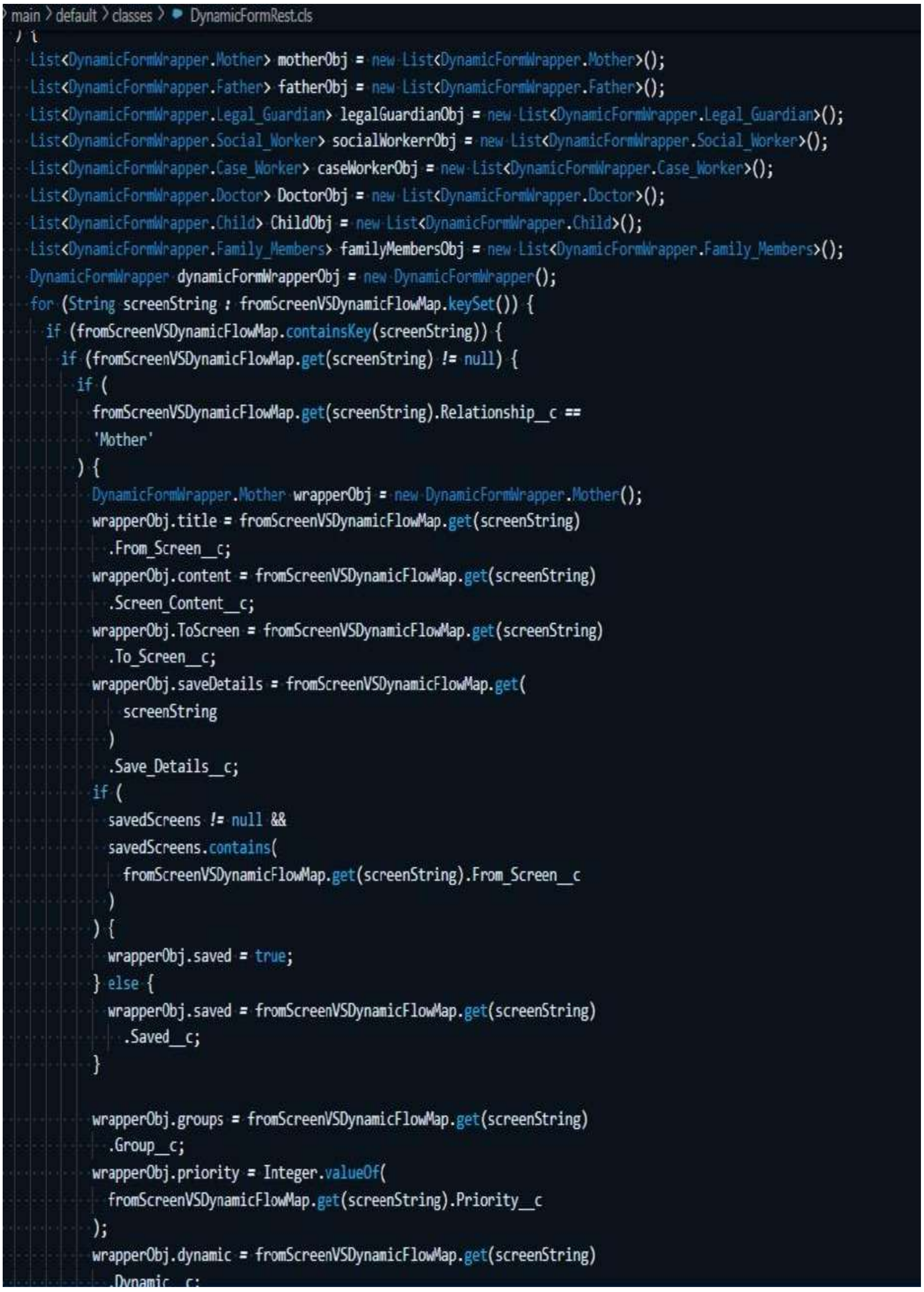
```
            if (caseResult[0].Fit_For_Wish__c != null) {
              if (caseResult[0].Fit_For_Wish__c == true) {
                dynamicFormWrapperObj.FitForWish = 'Yes';
              } else if (
                caseResult[0].Fit_For_Wish__c == false &&
                usedFields.contains('Fit_For_Wish__c')
              ) {
                dynamicFormWrapperObj.FitForWish = 'No';
              } else {
                dynamicFormWrapperObj.FitForWish = null;
              }
            }
            if (caseResult[0].Specification_Related_Wish__c != null) {
              dynamicFormWrapperObj.AnyWishSpecification = caseResult[0]
                .Specification_Related_Wish__c;
            }

            if (caseResult[0].Child_Condition_by_Doctor__c != null) {
              dynamicFormWrapperObj.ChildsCondition = caseResult[0]
                .Child_Condition_by_Doctor__c;
            }
          }
          dynamicFormWrapperObj.Family = FamilyList;
          dynamicFormWrapperObj.GroupList = groupSet;
          return dynamicFormWrapperObj;
        }
      }
    } catch (Exception e) {
      System.debug(
        'Exception: ' +
        e.getMessage() +
        ' At Line: ' +
        e.getLineNumber()
      );
    }
    return null;
  }

  @HttpPut
  global static void DynamicFormSaveDetails() {
    try {
      RestRequest request = RestContext.request;
      String xmlString = request.requestBody.toString();
      String applicationNo = RestContext.request.params.get('applicationNo');
      String formCompleted = RestContext.request.params.get('formCompleted');
      List<Case> caseList = new List<Case>();
```


efault > classes > DynamicFormRest.cls

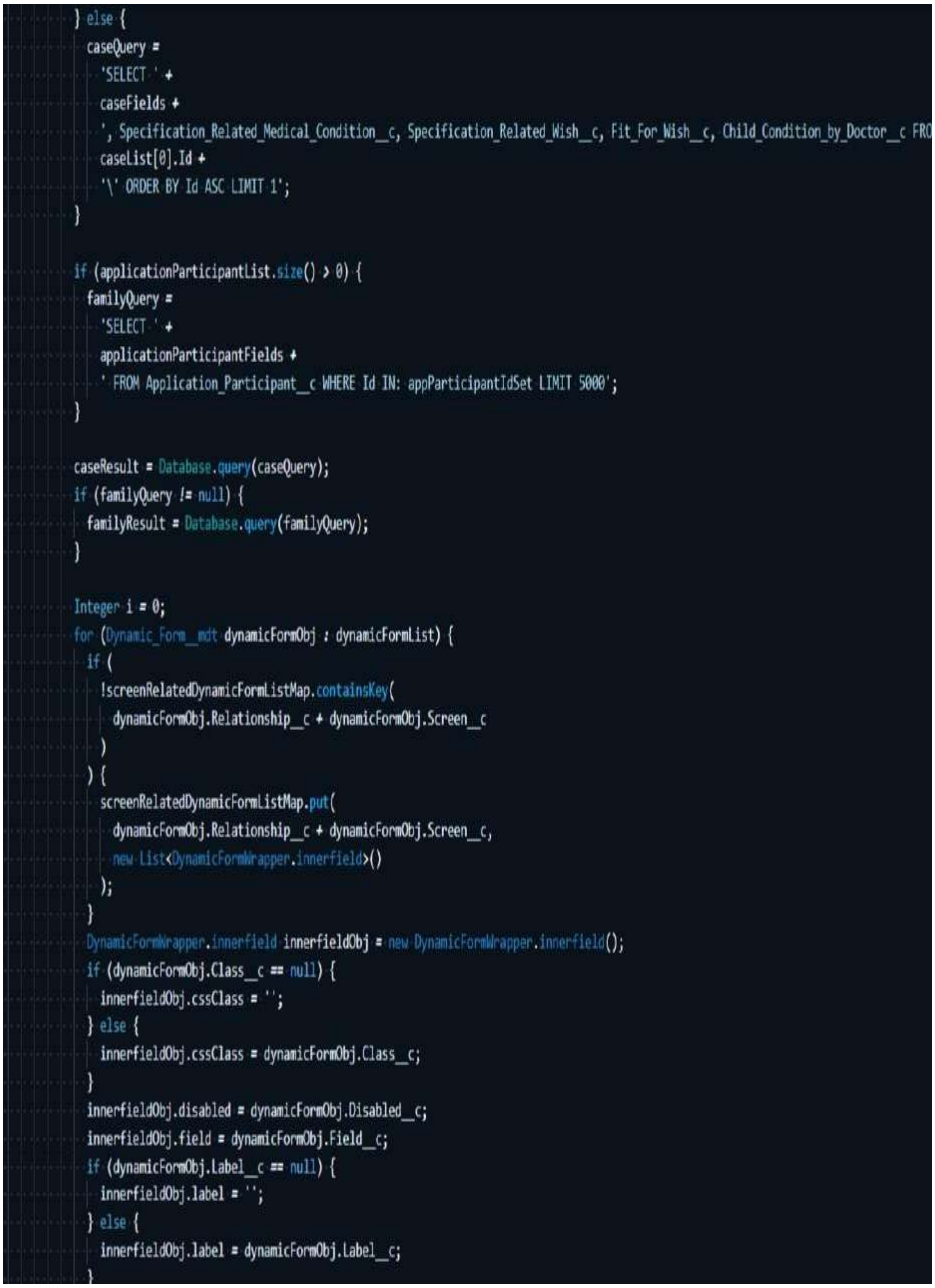
```
}

if (newCaseList.size() > 0) {
  if (newCaseList[0].Form_Completed__c == null) {
    newCaseList[0].Form_Completed__c = 0;
  }

  if (
    newCaseList[0].Form_Completed__c > 0 &&
    newCaseList[0].Form_Completed__c < 100
  ) {
    DynamicFormWrapper dynamicFormWrapper = DynamicFormRest.DynamicFormExistingDetails(
      applicationNumber,
      false,
      false
    );
    if (dynamicFormWrapper != null) {
      RestContext.response.responseBody = Blob.valueOf(
        '{"Status":"Success", "StatusCode":"200", "primaryContactName": "' +
        newCaseList[0].R_First_Name__c +
        '", "Record":' +
        System.JSON.serialize(dynamicFormWrapper) +
        '}'
      );
      RestContext.response.statusCode = 200;
    } else {
      RestContext.response.responseBody = Blob.valueOf(
        '{"Status":"Error", "StatusCode":"400"}'
      );
      RestContext.response.statusCode = 400;
    }
  } else if (newCaseList[0].Form_Completed__c == 100) {
    if (
      dynamicFormApplicationWrapperObj.consent == true &&
      dynamicFormApplicationWrapperObj.Dashboard == false
    ) {
      DynamicFormWrapper dynamicFormWrapper = DynamicFormRest.DynamicFormExistingDetails(
        applicationNumber,
        false,
        true
      );
      if (dynamicFormWrapper != null) {
        RestContext.response.responseBody = Blob.valueOf(
          '{"Status":"Success", "StatusCode":"200", "primaryContactFirstName": "' +
          newCaseList[0].R_First_Name__c +
          '", "primaryContactLastName": "' +
```



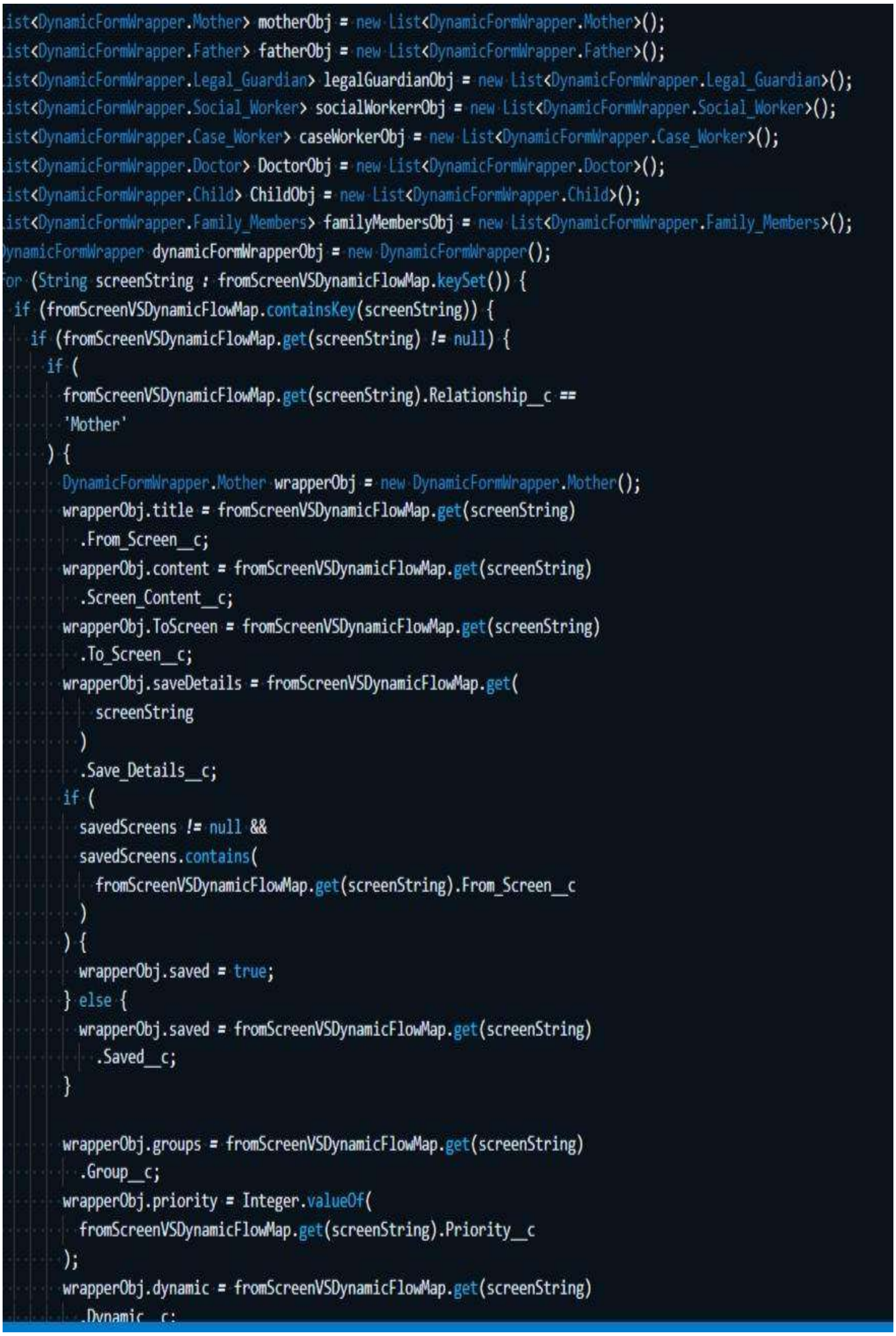
```
        newCaseObj.R_Address__c != null
    ) {
        newCaseObj.PC_First_Name__c = caseList[0].R_First_Name__c;
        newCaseObj.PC_Surname__c = caseList[0].R_Surname__c;
        newCaseObj.PC_Phone__c = newCaseObj.R_Phone__c;
        newCaseObj.PC_Email__c = newCaseObj.R_Email__c;
        newCaseObj.PC_Secondary_Phone__c = newCaseObj.R_Secondary_Phone__c;
        newCaseObj.PC_Address__c = newCaseObj.R_Address__c;
        newCaseObj.PC_State__c = newCaseObj.R_State__c;
        newCaseObj.PC_Postcode__c = newCaseObj.R_Postcode__c;
        newCaseObj.Relationship_to_child__c = newCaseObj.Referral_Relationship_to_Child__c;
        newCaseObj.ReferalContentDocumetId__c = newCaseObj.ParentContentDocumentId__c;
    }

    if (savedScreen.size() > 0)
        newCaseObj.Saved_Screen__c = String.join(
            new List<String>(savedScreen),
            ','
        );

    upsert newCaseObj;

    upsert newApplicationParticipantList;

    if (
        formCompleted != null &&
        formCompleted != '' &&
        String.isNotEmpty(formCompleted) &&
        String.isNotBlank(formCompleted) &&
        formCompleted == '100'
    ) {
        String emailId = newCaseObj.R_Email__c;
        SendEmailClass.sendEmailMethod1(
            'Application Form Completed',
            caseList[0],
            emailId
        );
        if (
            (socialWorkerContactObj != null &&
            socialWorkerContactObj != contactObj &&
            socialWorkerContactObj.Id != null &&
            socialWorkerContactObj.Email != null &&
            socialWorkerContactObj.Email != '' &&
            String.isNotEmpty(socialWorkerContactObj.Email) &&
            String.isNotBlank(socialWorkerContactObj.Email)) ||
            (doctorContactObj != null &&
```



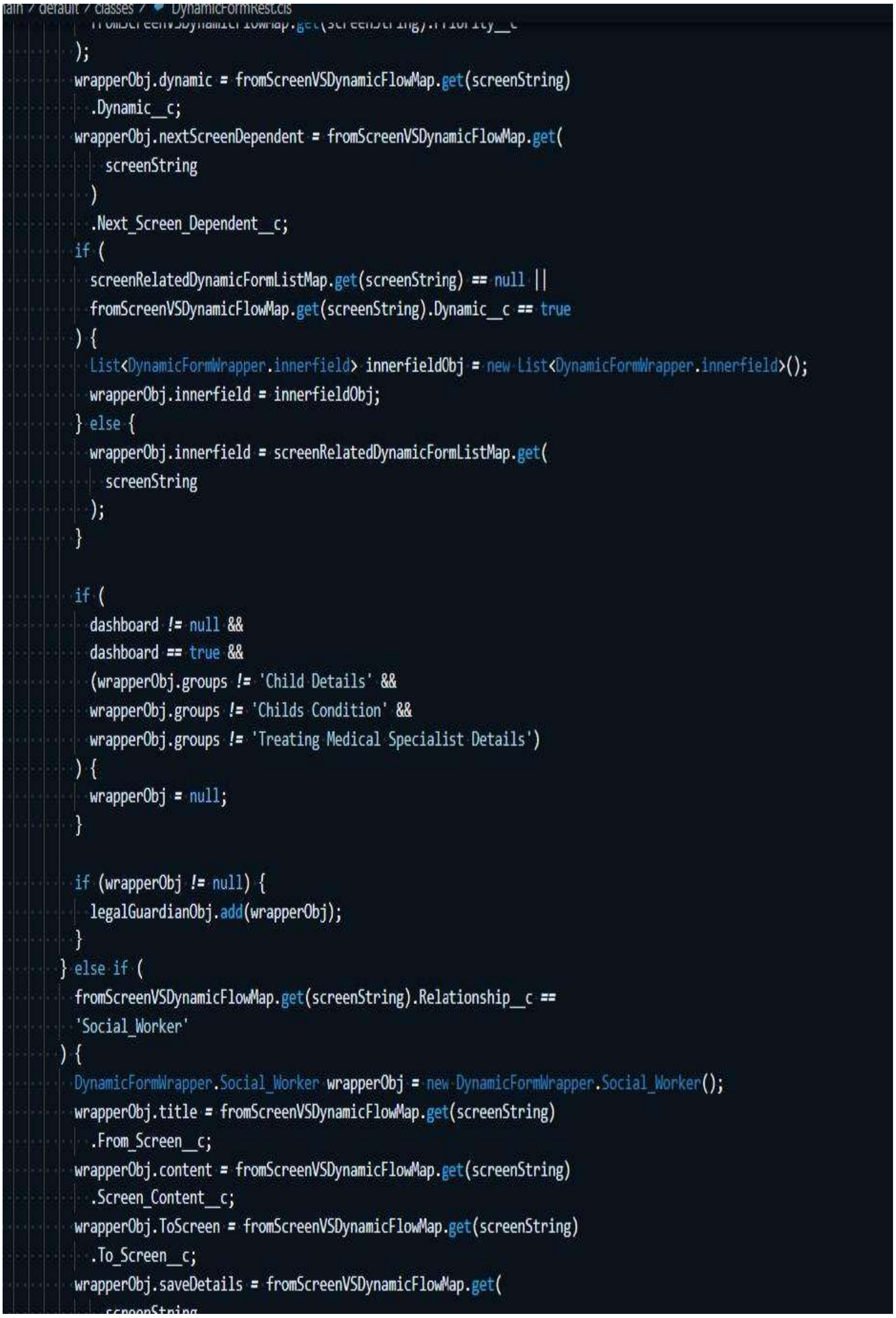
```
main > default > classes > DynamicFormRest.cls
) {
  List<DynamicFormWrapper.Mother> motherObj = new List<DynamicFormWrapper.Mother>();
  List<DynamicFormWrapper.Father> fatherObj = new List<DynamicFormWrapper.Father>();
  List<DynamicFormWrapper.Legal_Guardian> legalGuardianObj = new List<DynamicFormWrapper.Legal_Guardian>();
  List<DynamicFormWrapper.Social_Worker> socialWorkerrObj = new List<DynamicFormWrapper.Social_Worker>();
  List<DynamicFormWrapper.Case_Worker> caseWorkerObj = new List<DynamicFormWrapper.Case_Worker>();
  List<DynamicFormWrapper.Doctor> DoctorObj = new List<DynamicFormWrapper.Doctor>();
  List<DynamicFormWrapper.Child> ChildObj = new List<DynamicFormWrapper.Child>();
  List<DynamicFormWrapper.Family_Members> familyMembersObj = new List<DynamicFormWrapper.Family_Members>();
  DynamicFormWrapper dynamicFormWrapperObj = new DynamicFormWrapper();
  for (String screenString : fromScreenVSDynamicFlowMap.keySet()) {
    if (fromScreenVSDynamicFlowMap.containsKey(screenString)) {
      if (fromScreenVSDynamicFlowMap.get(screenString) != null) {
        if (
          fromScreenVSDynamicFlowMap.get(screenString).Relationship__c ==
          'Mother'
        ) {
          DynamicFormWrapper.Mother wrapperObj = new DynamicFormWrapper.Mother();
          wrapperObj.title = fromScreenVSDynamicFlowMap.get(screenString)
            .From_Screen__c;
          wrapperObj.content = fromScreenVSDynamicFlowMap.get(screenString)
            .Screen_Content__c;
          wrapperObj.ToScreen = fromScreenVSDynamicFlowMap.get(screenString)
            .To_Screen__c;
          wrapperObj.saveDetails = fromScreenVSDynamicFlowMap.get(
              screenString
            )
            .Save_Details__c;
          if (
            savedScreens != null &&
            savedScreens.contains(
              fromScreenVSDynamicFlowMap.get(screenString).From_Screen__c
            )
          ) {
            wrapperObj.saved = true;
          } else {
            wrapperObj.saved = fromScreenVSDynamicFlowMap.get(screenString)
              .Saved__c;
          }

          wrapperObj.groups = fromScreenVSDynamicFlowMap.get(screenString)
            .Group__c;
          wrapperObj.priority = Integer.valueOf(
            fromScreenVSDynamicFlowMap.get(screenString).Priority__c
          );
          wrapperObj.dynamic = fromScreenVSDynamicFlowMap.get(screenString)
            .Dynamic__c;
```



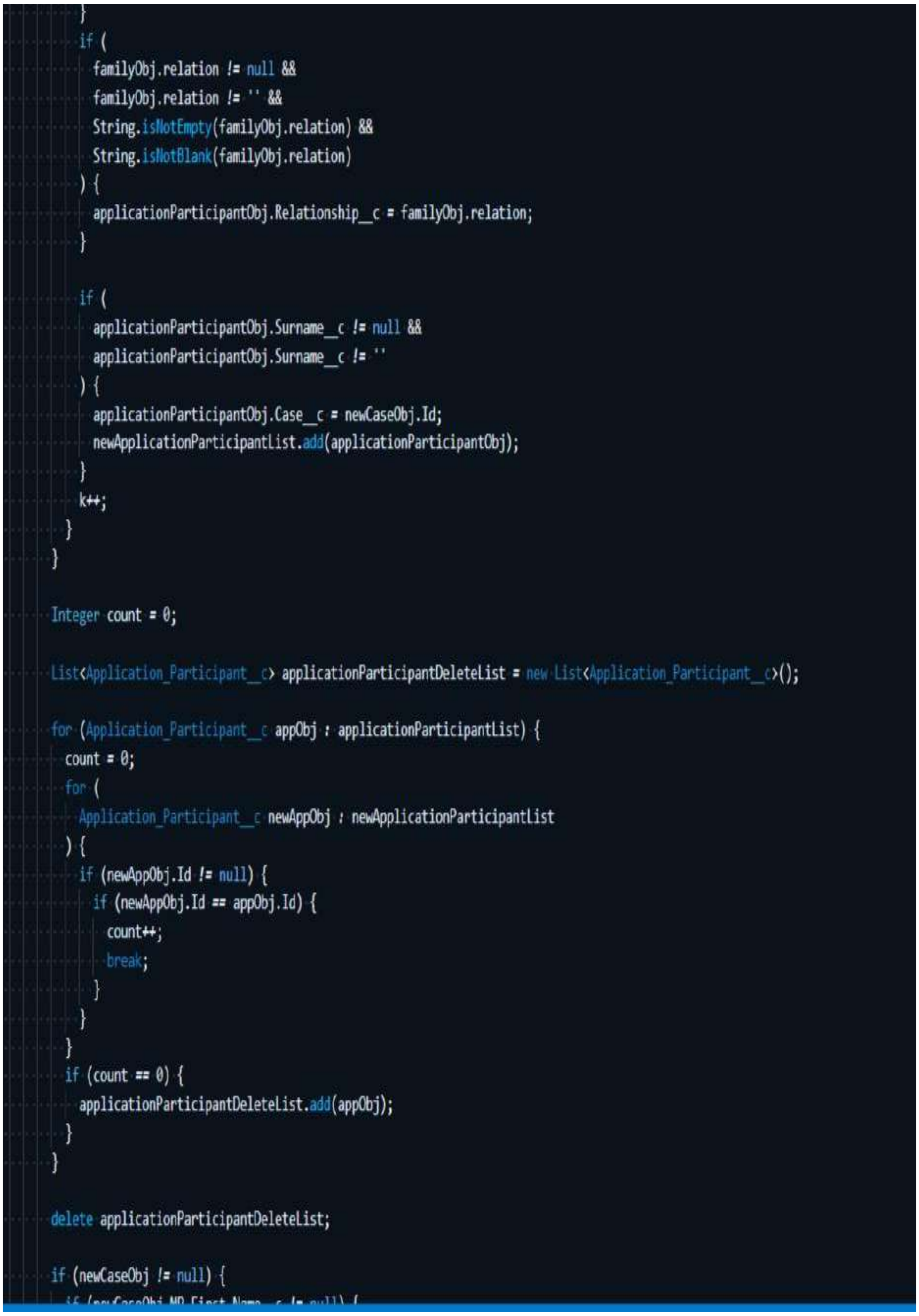
```
} else {
  caseQuery =
    'SELECT ' +
    caseFields +
    ', Specification_Related_Medical_Condition__c, Specification_Related_Wish__c, Fit_For_Wish__c, Child_Condition_by_Doctor__c FRO
    caseList[0].Id +
    '\' ORDER BY Id ASC LIMIT 1';
}

if (applicationParticipantList.size() > 0) {
  familyQuery =
    'SELECT ' +
    applicationParticipantFields +
    ' FROM Application_Participant__c WHERE Id IN: appParticipantIdSet LIMIT 5000';
}

caseResult = Database.query(caseQuery);
if (familyQuery != null) {
  familyResult = Database.query(familyQuery);
}

Integer i = 0;
for (Dynamic_Form__mdt dynamicFormObj : dynamicFormList) {
  if (
    !screenRelatedDynamicFormListMap.containsKey(
      dynamicFormObj.Relationship__c + dynamicFormObj.Screen__c
    )
  ) {
    screenRelatedDynamicFormListMap.put(
      dynamicFormObj.Relationship__c + dynamicFormObj.Screen__c,
      new List<DynamicFormWrapper.innerfield>()
    );
  }
  DynamicFormWrapper.innerfield innerfieldObj = new DynamicFormWrapper.innerfield();
  if (dynamicFormObj.Class__c == null) {
    innerfieldObj.cssClass = '';
  } else {
    innerfieldObj.cssClass = dynamicFormObj.Class__c;
  }
  innerfieldObj.disabled = dynamicFormObj.Disabled__c;
  innerfieldObj.field = dynamicFormObj.Field__c;
  if (dynamicFormObj.Label__c == null) {
    innerfieldObj.label = '';
  } else {
    innerfieldObj.label = dynamicFormObj.Label__c;
  }
```



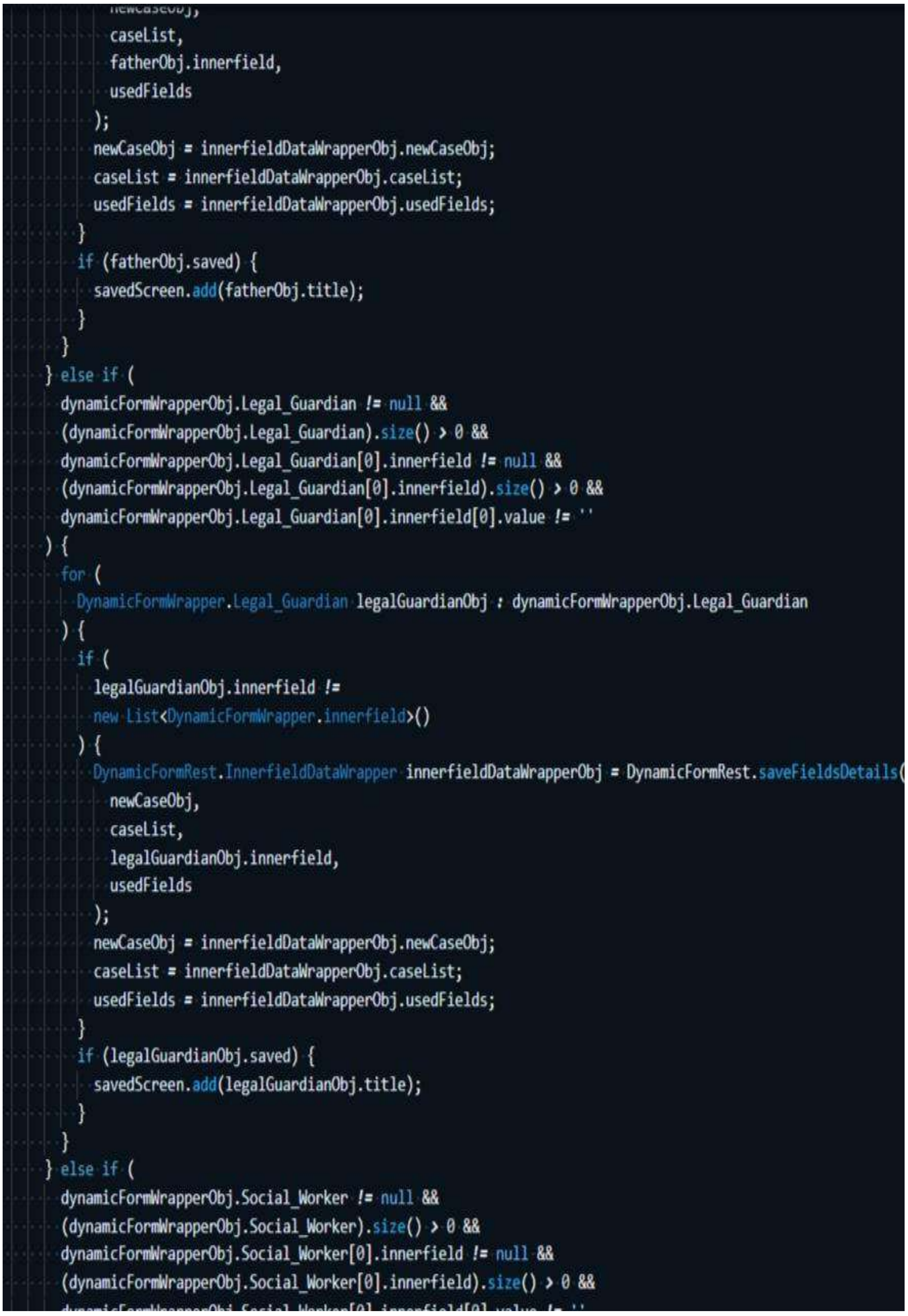
```
ist<DynamicFormWrapper.Mother> motherObj = new List<DynamicFormWrapper.Mother>();
ist<DynamicFormWrapper.Father> fatherObj = new List<DynamicFormWrapper.Father>();
ist<DynamicFormWrapper.Legal_Guardian> legalGuardianObj = new List<DynamicFormWrapper.Legal_Guardian>();
ist<DynamicFormWrapper.Social_Worker> socialWorkerrObj = new List<DynamicFormWrapper.Social_Worker>();
ist<DynamicFormWrapper.Case_Worker> caseWorkerObj = new List<DynamicFormWrapper.Case_Worker>();
ist<DynamicFormWrapper.Doctor> DoctorObj = new List<DynamicFormWrapper.Doctor>();
ist<DynamicFormWrapper.Child> ChildObj = new List<DynamicFormWrapper.Child>();
ist<DynamicFormWrapper.Family_Members> familyMembersObj = new List<DynamicFormWrapper.Family_Members>();
ynamicFormWrapper dynamicFormWrapperObj = new DynamicFormWrapper();
or (String screenString : fromScreenVSDynamicFlowMap.keySet()) {
  if (fromScreenVSDynamicFlowMap.containsKey(screenString)) {
    if (fromScreenVSDynamicFlowMap.get(screenString) != null) {
      if (
        fromScreenVSDynamicFlowMap.get(screenString).Relationship__c ==
        'Mother'
      ) {
        DynamicFormWrapper.Mother wrapperObj = new DynamicFormWrapper.Mother();
        wrapperObj.title = fromScreenVSDynamicFlowMap.get(screenString)
          .From_Screen__c;
        wrapperObj.content = fromScreenVSDynamicFlowMap.get(screenString)
          .Screen_Content__c;
        wrapperObj.ToScreen = fromScreenVSDynamicFlowMap.get(screenString)
          .To_Screen__c;
        wrapperObj.saveDetails = fromScreenVSDynamicFlowMap.get(
            screenString
          )
          .Save_Details__c;
        if (
          savedScreens != null &&
          savedScreens.contains(
            fromScreenVSDynamicFlowMap.get(screenString).From_Screen__c
          )
        ) {
          wrapperObj.saved = true;
        } else {
          wrapperObj.saved = fromScreenVSDynamicFlowMap.get(screenString)
            .Saved__c;
        }

        wrapperObj.groups = fromScreenVSDynamicFlowMap.get(screenString)
          .Group__c;
        wrapperObj.priority = Integer.valueOf(
          fromScreenVSDynamicFlowMap.get(screenString).Priority__c
        );
        wrapperObj.dynamic = fromScreenVSDynamicFlowMap.get(screenString)
          .Dynamic__c;
```



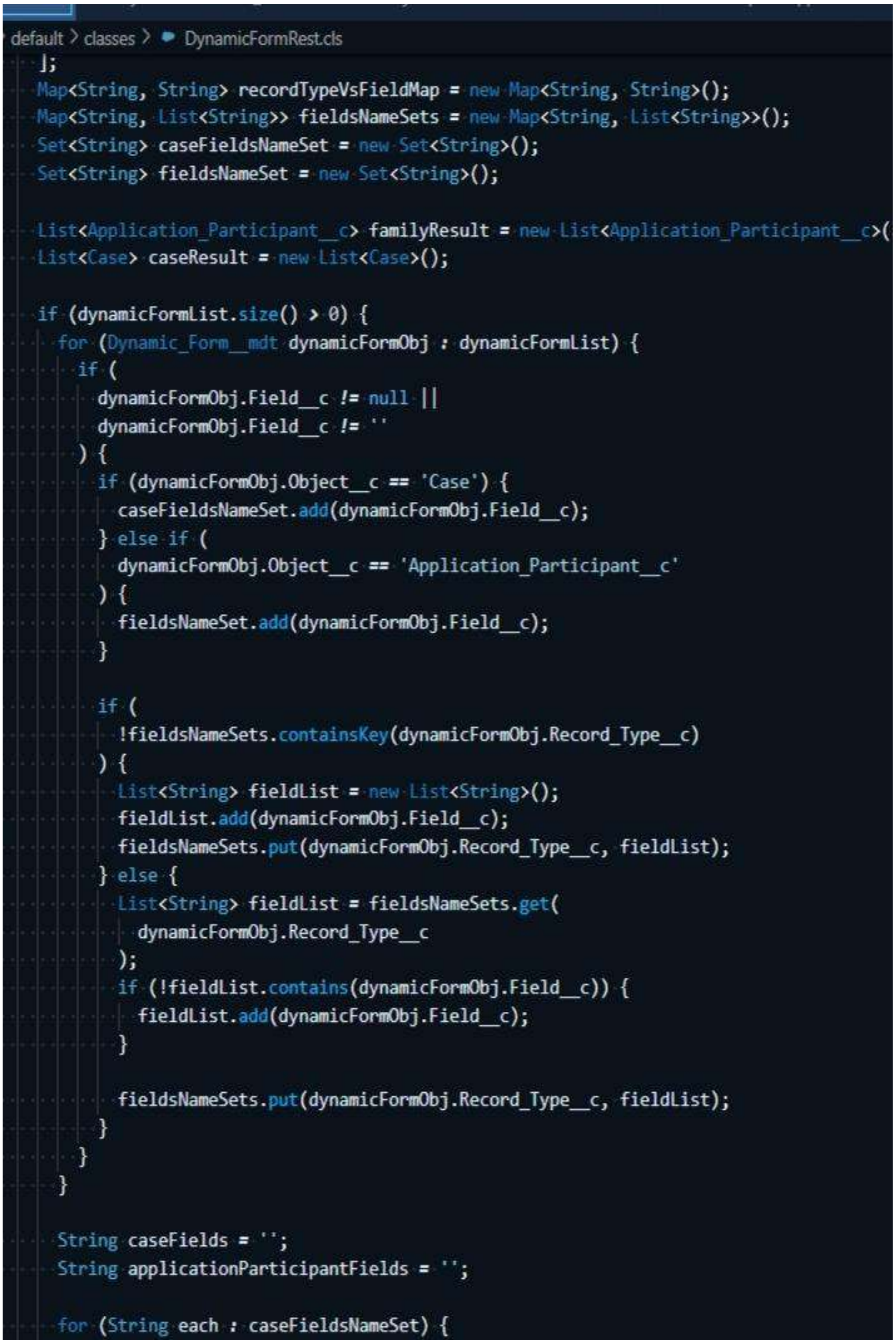
```
ain / default / classes / DynamicFormRest.cls
      fromScreenVSDynamicFlowMap.get(screenString).Priority__c
    );
    wrapperObj.dynamic = fromScreenVSDynamicFlowMap.get(screenString)
      .Dynamic__c;
    wrapperObj.nextScreenDependent = fromScreenVSDynamicFlowMap.get(
        screenString
      )
      .Next_Screen_Dependent__c;
    if (
      screenRelatedDynamicFormListMap.get(screenString) == null ||
      fromScreenVSDynamicFlowMap.get(screenString).Dynamic__c == true
    ) {
      List<DynamicFormWrapper.innerfield> innerfieldObj = new List<DynamicFormWrapper.innerfield>();
      wrapperObj.innerfield = innerfieldObj;
    } else {
      wrapperObj.innerfield = screenRelatedDynamicFormListMap.get(
        screenString
      );
    }

    if (
      dashboard != null &&
      dashboard == true &&
      (wrapperObj.groups != 'Child Details' &&
      wrapperObj.groups != 'Childs Condition' &&
      wrapperObj.groups != 'Treating Medical Specialist Details')
    ) {
      wrapperObj = null;
    }

    if (wrapperObj != null) {
      legalGuardianObj.add(wrapperObj);
    }
  } else if (
    fromScreenVSDynamicFlowMap.get(screenString).Relationship__c ==
    'Social_Worker'
  ) {
    DynamicFormWrapper.Social_Worker wrapperObj = new DynamicFormWrapper.Social_Worker();
    wrapperObj.title = fromScreenVSDynamicFlowMap.get(screenString)
      .From_Screen__c;
    wrapperObj.content = fromScreenVSDynamicFlowMap.get(screenString)
      .Screen_Content__c;
    wrapperObj.ToScreen = fromScreenVSDynamicFlowMap.get(screenString)
      .To_Screen__c;
    wrapperObj.saveDetails = fromScreenVSDynamicFlowMap.get(
        screenString
```



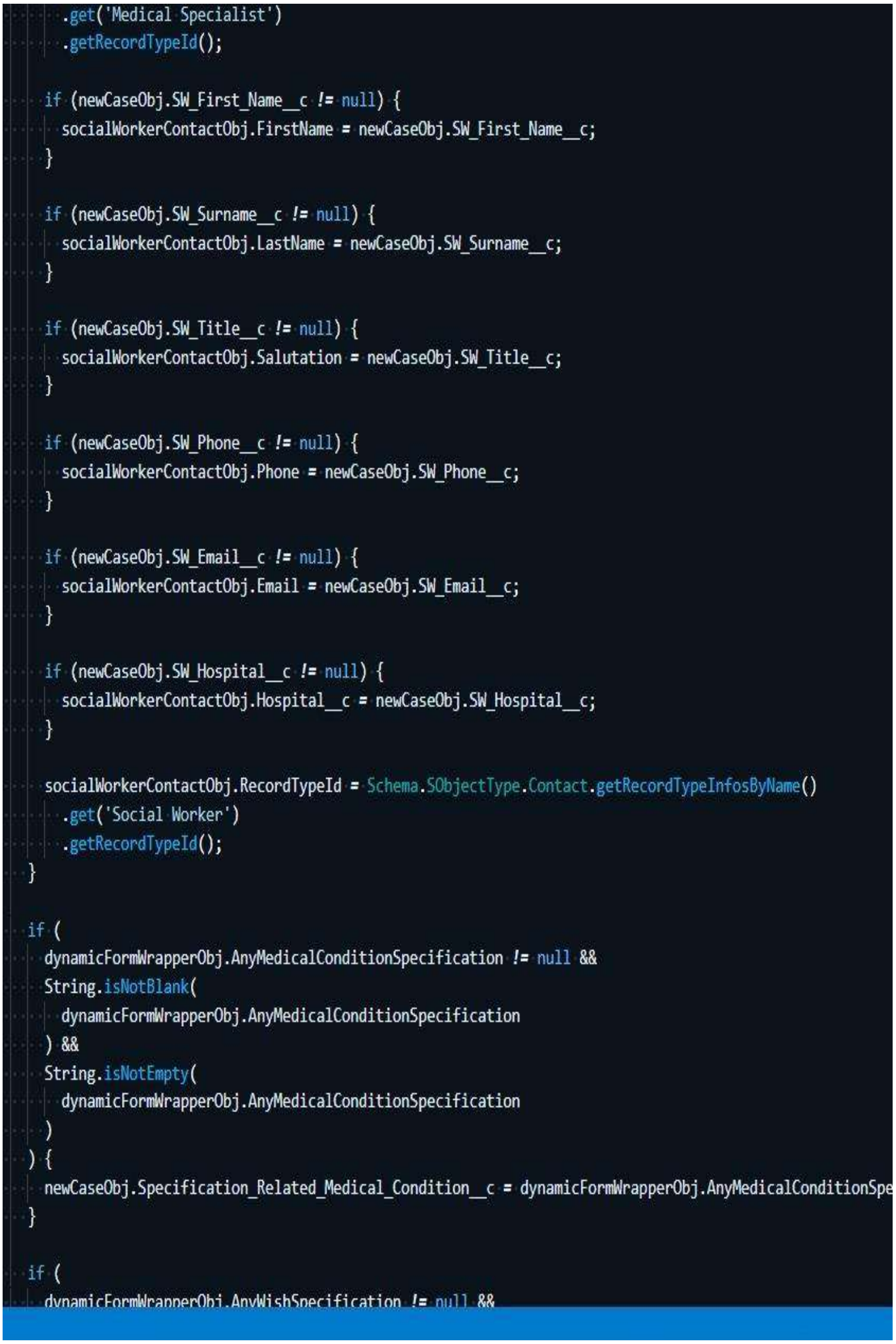
```
    }
    if (
      familyObj.relation != null &&
      familyObj.relation != '' &&
      String.isNotEmpty(familyObj.relation) &&
      String.isNotBlank(familyObj.relation)
    ) {
      applicationParticipantObj.Relationship__c = familyObj.relation;
    }

    if (
      applicationParticipantObj.Surname__c != null &&
      applicationParticipantObj.Surname__c != ''
    ) {
      applicationParticipantObj.Case__c = newCaseObj.Id;
      newApplicationParticipantList.add(applicationParticipantObj);
    }
    k++;
  }
}

Integer count = 0;

List<Application_Participant__c> applicationParticipantDeleteList = new List<Application_Participant__c>();

for (Application_Participant__c appObj : applicationParticipantList) {
  count = 0;
  for (
    Application_Participant__c newAppObj : newApplicationParticipantList
  ) {
    if (newAppObj.Id != null) {
      if (newAppObj.Id == appObj.Id) {
        count++;
        break;
      }
    }
  }
  if (count == 0) {
    applicationParticipantDeleteList.add(appObj);
  }
}

delete applicationParticipantDeleteList;

if (newCaseObj != null) {
  if (newCaseObj.MP_First_Name__c != null) {
```



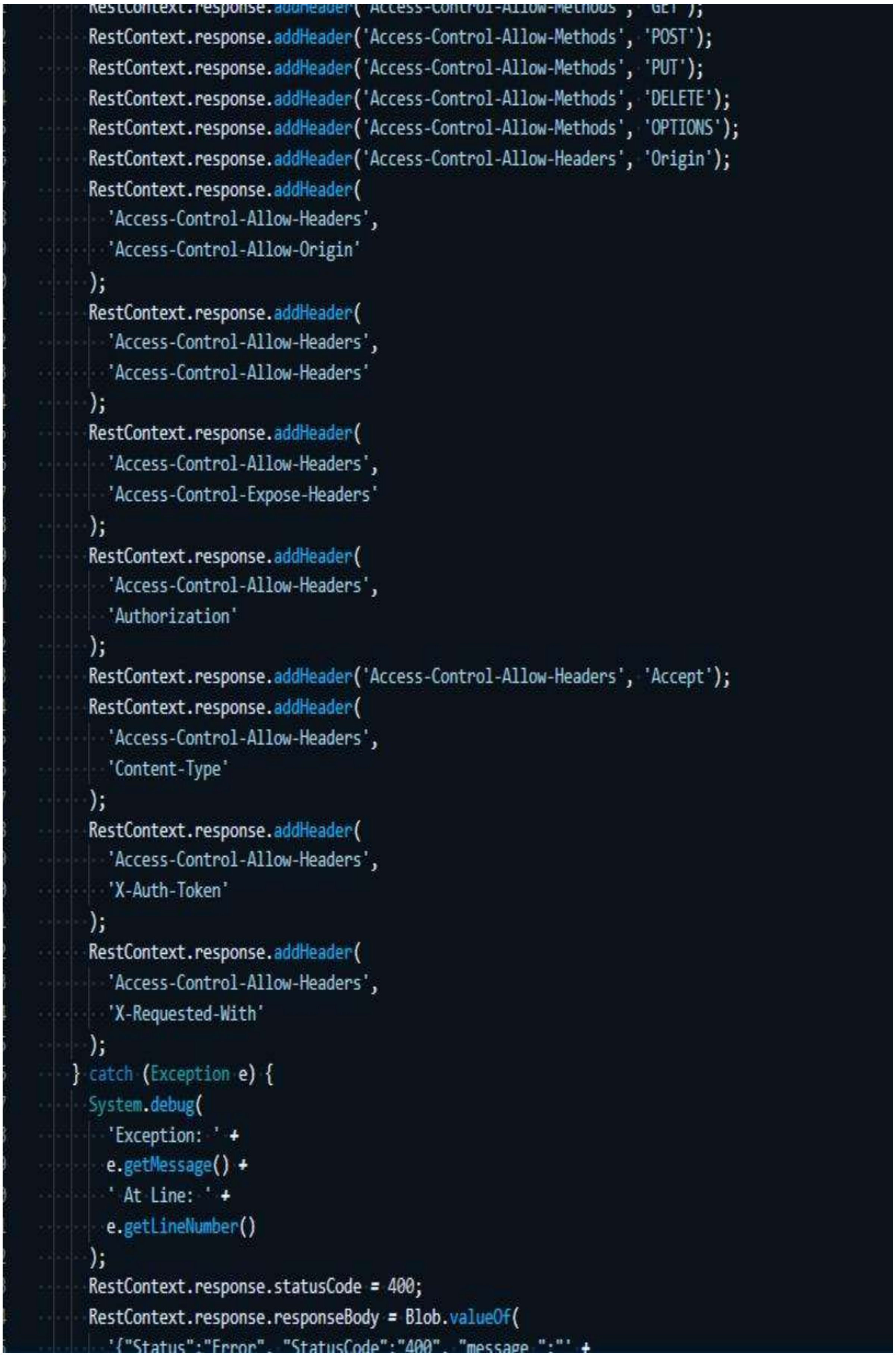
```
            newCaseObj,
            caseList,
            fatherObj.innerfield,
            usedFields
          );
          newCaseObj = innerfieldDataWrapperObj.newCaseObj;
          caseList = innerfieldDataWrapperObj.caseList;
          usedFields = innerfieldDataWrapperObj.usedFields;
        }
        if (fatherObj.saved) {
          savedScreen.add(fatherObj.title);
        }
      }
    } else if (
      dynamicFormWrapperObj.Legal_Guardian != null &&
      (dynamicFormWrapperObj.Legal_Guardian).size() > 0 &&
      dynamicFormWrapperObj.Legal_Guardian[0].innerfield != null &&
      (dynamicFormWrapperObj.Legal_Guardian[0].innerfield).size() > 0 &&
      dynamicFormWrapperObj.Legal_Guardian[0].innerfield[0].value != ''
    ) {
      for (
        DynamicFormWrapper.Legal_Guardian legalGuardianObj : dynamicFormWrapperObj.Legal_Guardian
      ) {
        if (
          legalGuardianObj.innerfield !=
          new List<DynamicFormWrapper.innerfield>()
        ) {
          DynamicFormRest.InnerfieldDataWrapper innerfieldDataWrapperObj = DynamicFormRest.saveFieldsDetails(
            newCaseObj,
            caseList,
            legalGuardianObj.innerfield,
            usedFields
          );
          newCaseObj = innerfieldDataWrapperObj.newCaseObj;
          caseList = innerfieldDataWrapperObj.caseList;
          usedFields = innerfieldDataWrapperObj.usedFields;
        }
        if (legalGuardianObj.saved) {
          savedScreen.add(legalGuardianObj.title);
        }
      }
    } else if (
      dynamicFormWrapperObj.Social_Worker != null &&
      (dynamicFormWrapperObj.Social_Worker).size() > 0 &&
      dynamicFormWrapperObj.Social_Worker[0].innerfield != null &&
      (dynamicFormWrapperObj.Social_Worker[0].innerfield).size() > 0 &&
      dynamicFormWrapperObj.Social_Worker[0].innerfield[0].value != ''
```


default > classes > DynamicFormRest.cls

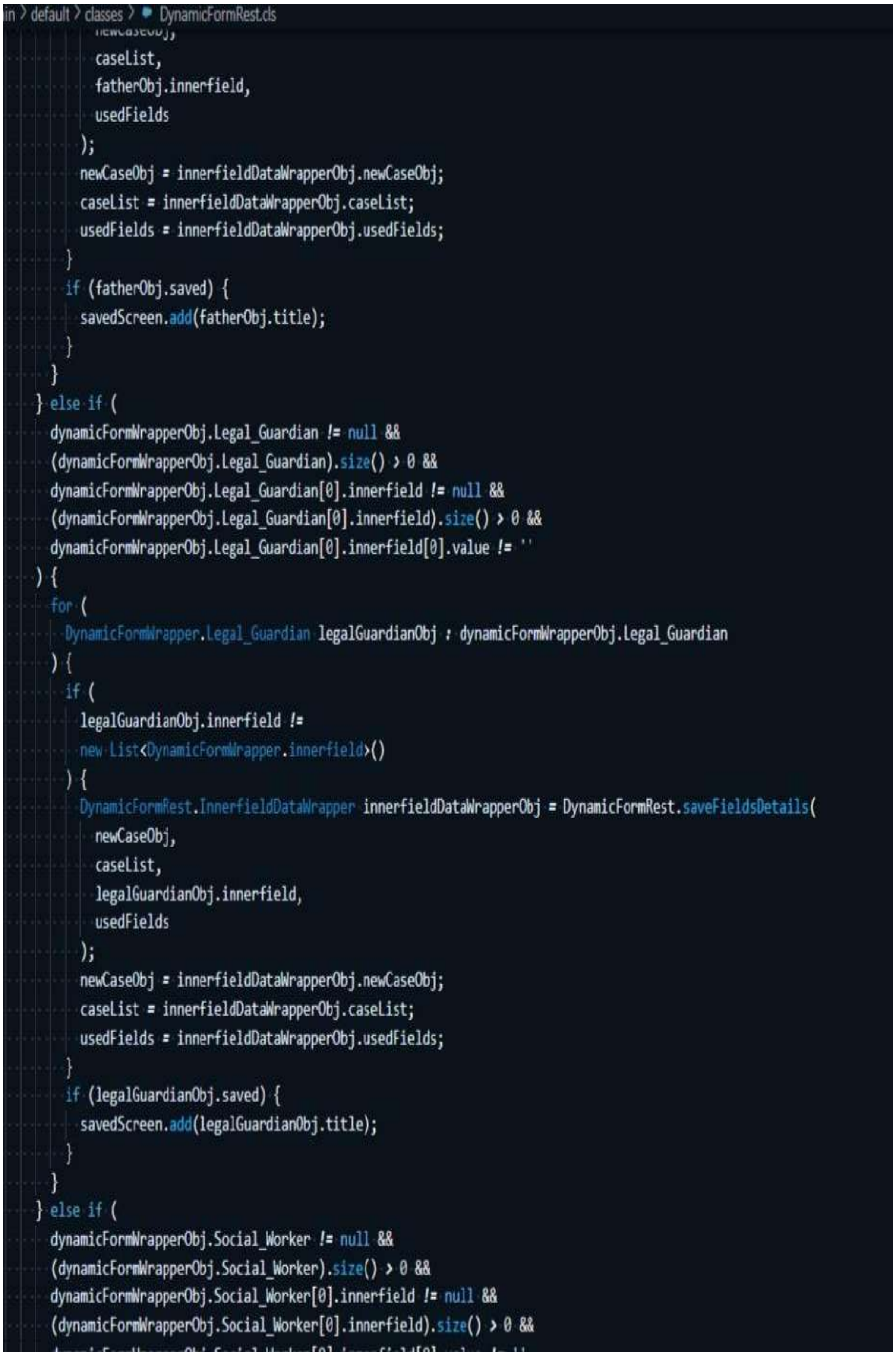
```
    ];
    Map<String, String> recordTypeVsFieldMap = new Map<String, String>();
    Map<String, List<String>> fieldsNameSets = new Map<String, List<String>>();
    Set<String> caseFieldsNameSet = new Set<String>();
    Set<String> fieldsNameSet = new Set<String>();

    List<Application_Participant__c> familyResult = new List<Application_Participant__c>(
    List<Case> caseResult = new List<Case>();

    if (dynamicFormList.size() > 0) {
      for (Dynamic_Form__mdt dynamicFormObj : dynamicFormList) {
        if (
          dynamicFormObj.Field__c != null ||
          dynamicFormObj.Field__c != ''
        ) {
          if (dynamicFormObj.Object__c == 'Case') {
            caseFieldsNameSet.add(dynamicFormObj.Field__c);
          } else if (
            dynamicFormObj.Object__c == 'Application_Participant__c'
          ) {
            fieldsNameSet.add(dynamicFormObj.Field__c);
          }

          if (
            !fieldsNameSets.containsKey(dynamicFormObj.Record_Type__c)
          ) {
            List<String> fieldList = new List<String>();
            fieldList.add(dynamicFormObj.Field__c);
            fieldsNameSets.put(dynamicFormObj.Record_Type__c, fieldList);
          } else {
            List<String> fieldList = fieldsNameSets.get(
              dynamicFormObj.Record_Type__c
            );
            if (!fieldList.contains(dynamicFormObj.Field__c)) {
              fieldList.add(dynamicFormObj.Field__c);
            }

            fieldsNameSets.put(dynamicFormObj.Record_Type__c, fieldList);
          }
        }
      }

      String caseFields = '';
      String applicationParticipantFields = '';

      for (String each : caseFieldsNameSet) {
```



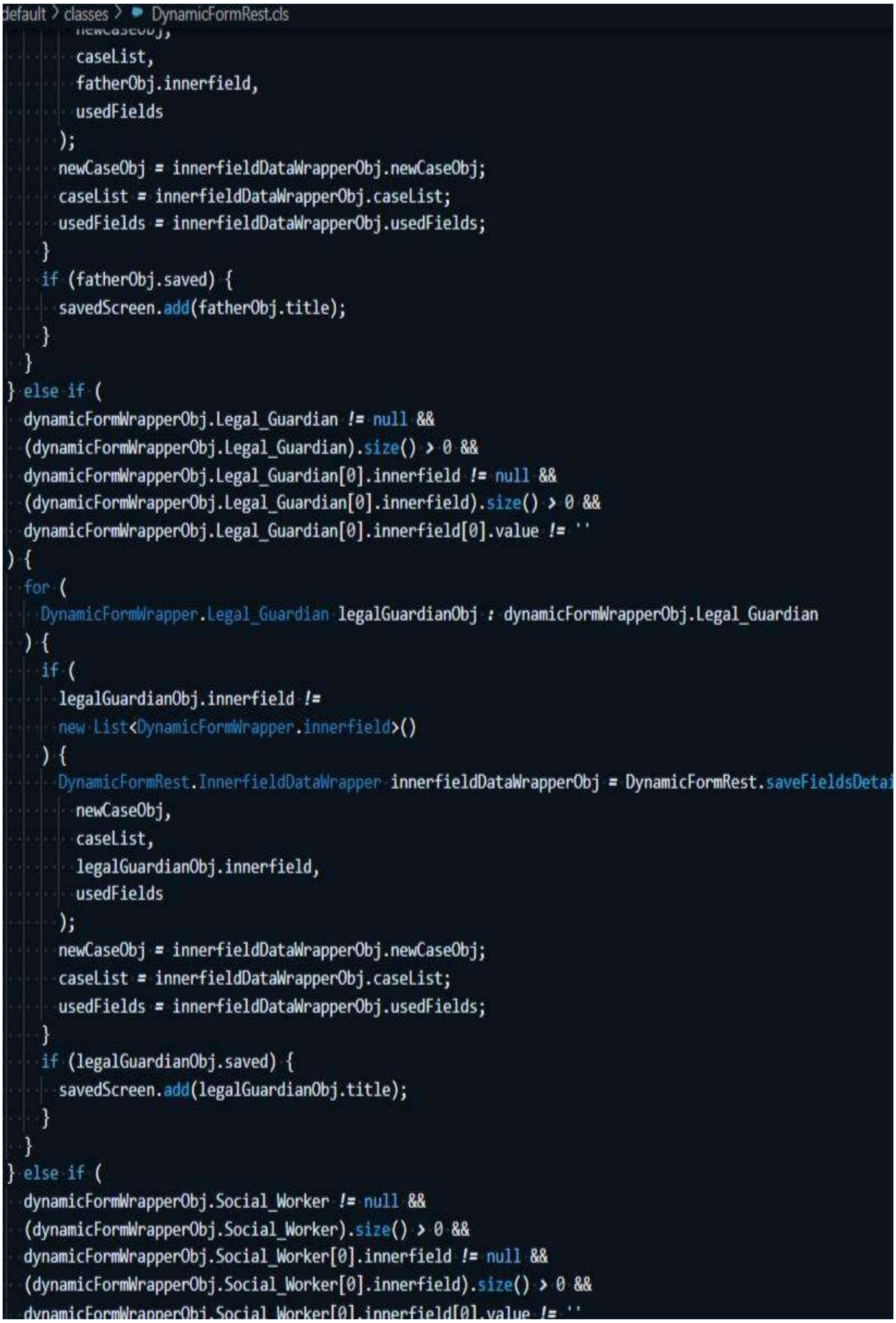
```
        .get('Medical Specialist')
        .getRecordTypeId();

    if (newCaseObj.SW_First_Name__c != null) {
      socialWorkerContactObj.FirstName = newCaseObj.SW_First_Name__c;
    }

    if (newCaseObj.SW_Surname__c != null) {
      socialWorkerContactObj.LastName = newCaseObj.SW_Surname__c;
    }

    if (newCaseObj.SW_Title__c != null) {
      socialWorkerContactObj.Salutation = newCaseObj.SW_Title__c;
    }

    if (newCaseObj.SW_Phone__c != null) {
      socialWorkerContactObj.Phone = newCaseObj.SW_Phone__c;
    }

    if (newCaseObj.SW_Email__c != null) {
      socialWorkerContactObj.Email = newCaseObj.SW_Email__c;
    }

    if (newCaseObj.SW_Hospital__c != null) {
      socialWorkerContactObj.Hospital__c = newCaseObj.SW_Hospital__c;
    }

    socialWorkerContactObj.RecordTypeId = Schema.SObjectType.Contact.getRecordTypeInfosByName()
      .get('Social Worker')
      .getRecordTypeId();
  }

  if (
    dynamicFormWrapperObj.AnyMedicalConditionSpecification != null &&
    String.isNotBlank(
      dynamicFormWrapperObj.AnyMedicalConditionSpecification
    ) &&
    String.isNotEmpty(
      dynamicFormWrapperObj.AnyMedicalConditionSpecification
    )
  ) {
    newCaseObj.Specification_Related_Medical_Condition__c = dynamicFormWrapperObj.AnyMedicalConditionSpe
  }

  if (
    dynamicFormWrapperObj.AnyWishSpecification != null &&
```



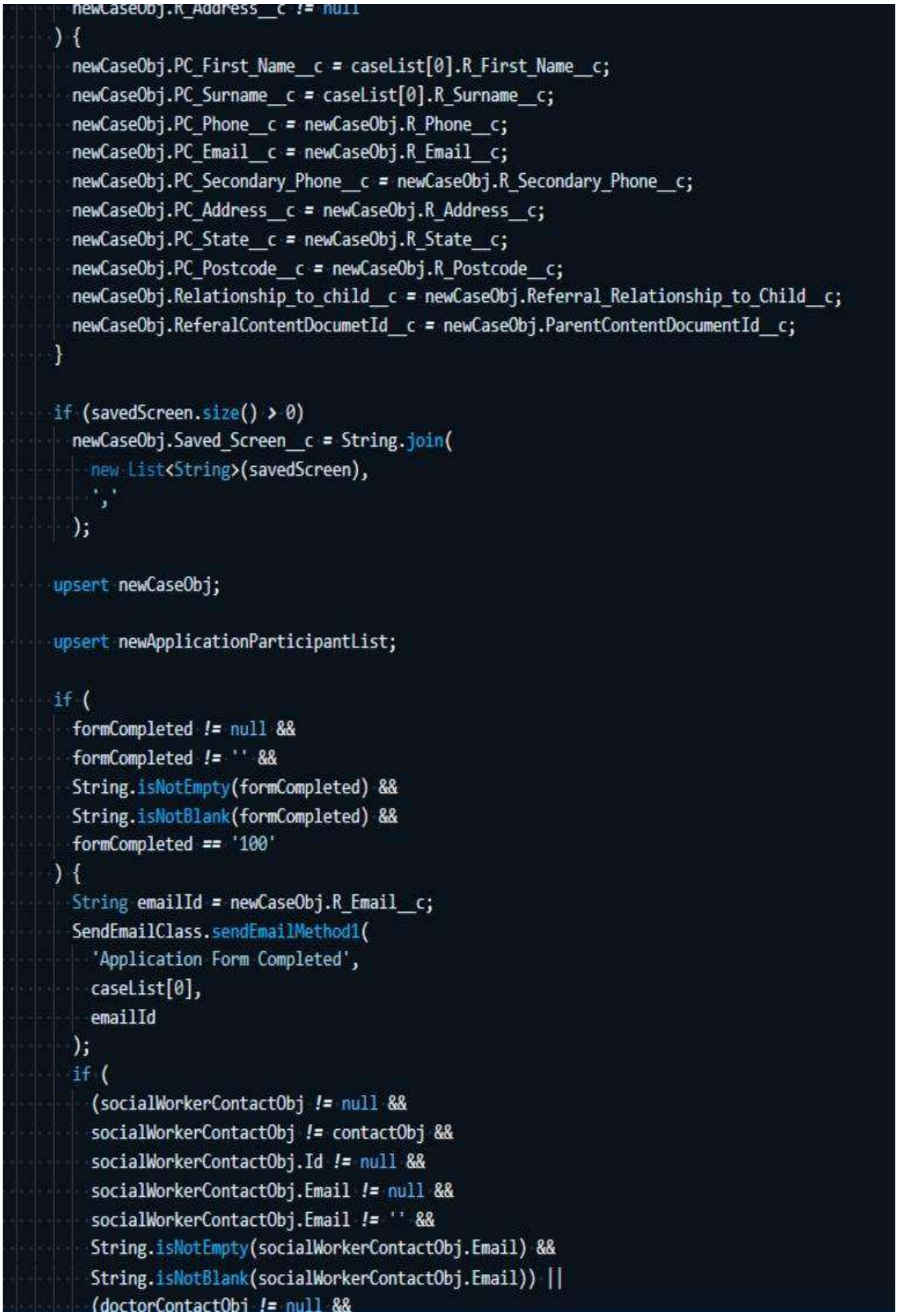
```
        RestContext.response.addHeader('Access-Control-Allow-Methods', 'GET');
        RestContext.response.addHeader('Access-Control-Allow-Methods', 'POST');
        RestContext.response.addHeader('Access-Control-Allow-Methods', 'PUT');
        RestContext.response.addHeader('Access-Control-Allow-Methods', 'DELETE');
        RestContext.response.addHeader('Access-Control-Allow-Methods', 'OPTIONS');
        RestContext.response.addHeader('Access-Control-Allow-Headers', 'Origin');
        RestContext.response.addHeader(
          'Access-Control-Allow-Headers',
          'Access-Control-Allow-Origin'
        );
        RestContext.response.addHeader(
          'Access-Control-Allow-Headers',
          'Access-Control-Allow-Headers'
        );
        RestContext.response.addHeader(
          'Access-Control-Allow-Headers',
          'Access-Control-Expose-Headers'
        );
        RestContext.response.addHeader(
          'Access-Control-Allow-Headers',
          'Authorization'
        );
        RestContext.response.addHeader('Access-Control-Allow-Headers', 'Accept');
        RestContext.response.addHeader(
          'Access-Control-Allow-Headers',
          'Content-Type'
        );
        RestContext.response.addHeader(
          'Access-Control-Allow-Headers',
          'X-Auth-Token'
        );
        RestContext.response.addHeader(
          'Access-Control-Allow-Headers',
          'X-Requested-With'
        );
      } catch (Exception e) {
        System.debug(
          'Exception: ' +
          e.getMessage() +
          ' At Line: ' +
          e.getLineNumber()
        );
        RestContext.response.statusCode = 400;
        RestContext.response.responseBody = Blob.valueOf(
          '{"Status":"Error", "StatusCode":"400", "message_":"' +
```



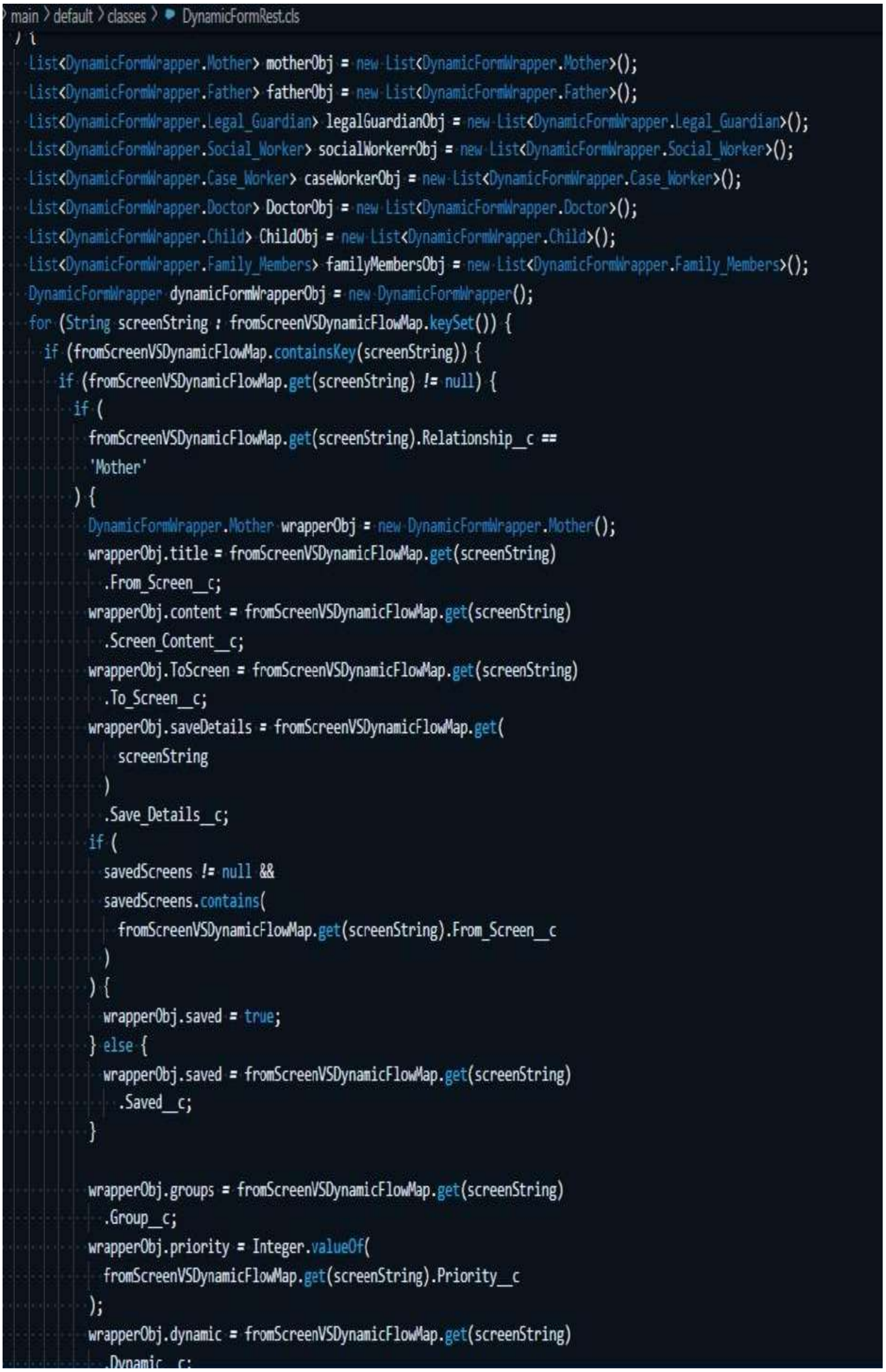
```
in > default > classes > DynamicFormRest.cls
        newCaseObj,
        caseList,
        fatherObj.innerfield,
        usedFields
      );
      newCaseObj = innerfieldDataWrapperObj.newCaseObj;
      caseList = innerfieldDataWrapperObj.caseList;
      usedFields = innerfieldDataWrapperObj.usedFields;
    }
    if (fatherObj.saved) {
      savedScreen.add(fatherObj.title);
    }
  }
} else if (
  dynamicFormWrapperObj.Legal_Guardian != null &&
  (dynamicFormWrapperObj.Legal_Guardian).size() > 0 &&
  dynamicFormWrapperObj.Legal_Guardian[0].innerfield != null &&
  (dynamicFormWrapperObj.Legal_Guardian[0].innerfield).size() > 0 &&
  dynamicFormWrapperObj.Legal_Guardian[0].innerfield[0].value != ''
) {
  for (
    DynamicFormWrapper.Legal_Guardian legalGuardianObj : dynamicFormWrapperObj.Legal_Guardian
  ) {
    if (
      legalGuardianObj.innerfield !=
      new List<DynamicFormWrapper.innerfield>()
    ) {
      DynamicFormRest.InnerfieldDataWrapper innerfieldDataWrapperObj = DynamicFormRest.saveFieldsDetails(
        newCaseObj,
        caseList,
        legalGuardianObj.innerfield,
        usedFields
      );
      newCaseObj = innerfieldDataWrapperObj.newCaseObj;
      caseList = innerfieldDataWrapperObj.caseList;
      usedFields = innerfieldDataWrapperObj.usedFields;
    }
    if (legalGuardianObj.saved) {
      savedScreen.add(legalGuardianObj.title);
    }
  }
} else if (
  dynamicFormWrapperObj.Social_Worker != null &&
  (dynamicFormWrapperObj.Social_Worker).size() > 0 &&
  dynamicFormWrapperObj.Social_Worker[0].innerfield != null &&
  (dynamicFormWrapperObj.Social_Worker[0].innerfield).size() > 0 &&
```



```
default > classes > DynamicFormRest.cls
        newCaseObj,
        caseList,
        fatherObj.innerfield,
        usedFields
      );
      newCaseObj = innerfieldDataWrapperObj.newCaseObj;
      caseList = innerfieldDataWrapperObj.caseList;
      usedFields = innerfieldDataWrapperObj.usedFields;
    }
    if (fatherObj.saved) {
      savedScreen.add(fatherObj.title);
    }
  }
} else if (
  dynamicFormWrapperObj.Legal_Guardian != null &&
  (dynamicFormWrapperObj.Legal_Guardian).size() > 0 &&
  dynamicFormWrapperObj.Legal_Guardian[0].innerfield != null &&
  (dynamicFormWrapperObj.Legal_Guardian[0].innerfield).size() > 0 &&
  dynamicFormWrapperObj.Legal_Guardian[0].innerfield[0].value != ''
) {
  for (
    DynamicFormWrapper.Legal_Guardian legalGuardianObj : dynamicFormWrapperObj.Legal_Guardian
  ) {
    if (
      legalGuardianObj.innerfield !=
      new List<DynamicFormWrapper.innerfield>()
    ) {
      DynamicFormRest.InnerfieldDataWrapper innerfieldDataWrapperObj = DynamicFormRest.saveFieldsDetai
        newCaseObj,
        caseList,
        legalGuardianObj.innerfield,
        usedFields
      );
      newCaseObj = innerfieldDataWrapperObj.newCaseObj;
      caseList = innerfieldDataWrapperObj.caseList;
      usedFields = innerfieldDataWrapperObj.usedFields;
    }
    if (legalGuardianObj.saved) {
      savedScreen.add(legalGuardianObj.title);
    }
  }
} else if (
  dynamicFormWrapperObj.Social_Worker != null &&
  (dynamicFormWrapperObj.Social_Worker).size() > 0 &&
  dynamicFormWrapperObj.Social_Worker[0].innerfield != null &&
  (dynamicFormWrapperObj.Social_Worker[0].innerfield).size() > 0 &&
  dynamicFormWrapperObj.Social_Worker[0].innerfield[0].value != ''
```

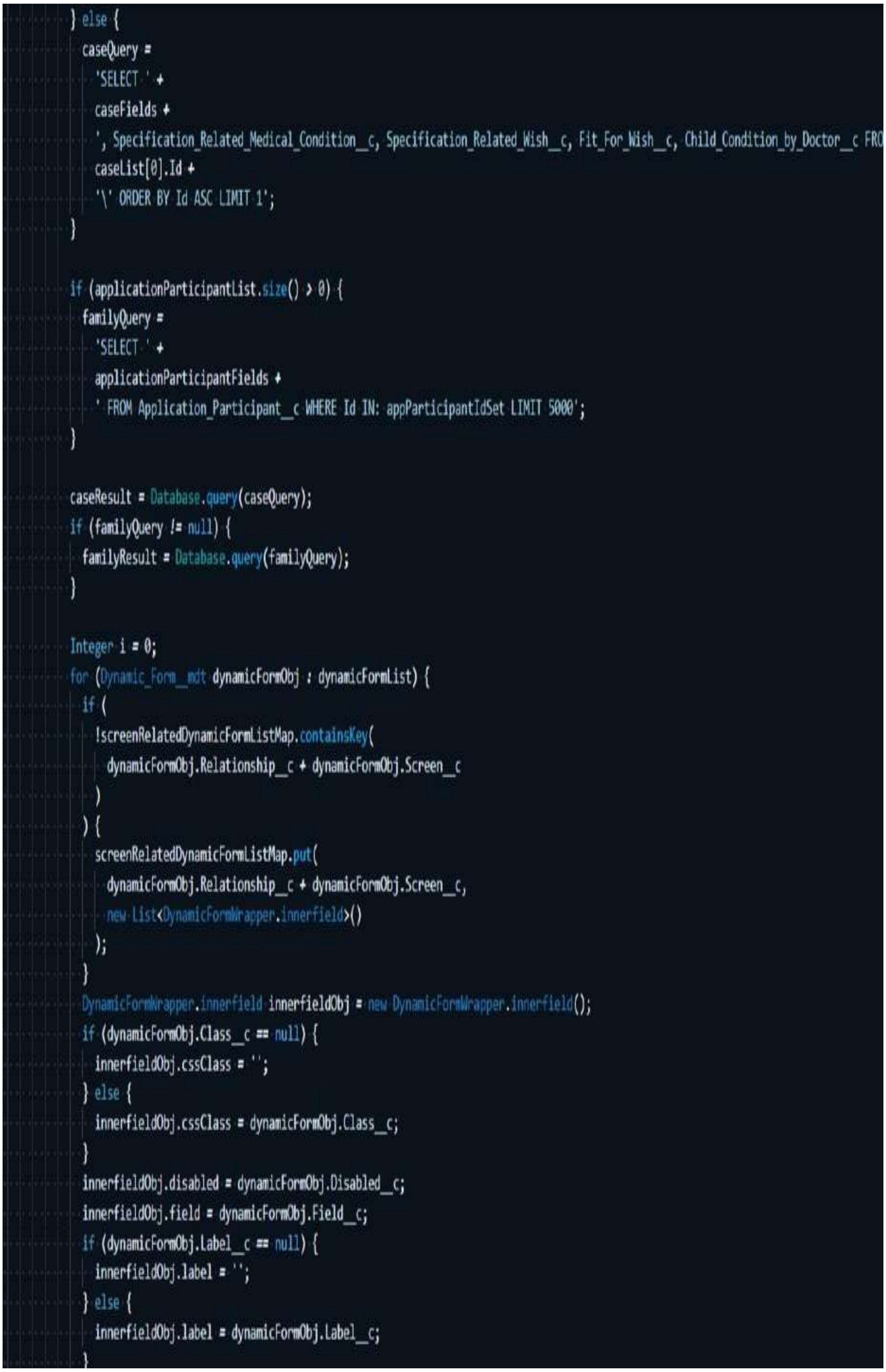


```
      newCaseObj.R_Address__c != null
    ) {
      newCaseObj.PC_First_Name__c = caseList[0].R_First_Name__c;
      newCaseObj.PC_Surname__c = caseList[0].R_Surname__c;
      newCaseObj.PC_Phone__c = newCaseObj.R_Phone__c;
      newCaseObj.PC_Email__c = newCaseObj.R_Email__c;
      newCaseObj.PC_Secondary_Phone__c = newCaseObj.R_Secondary_Phone__c;
      newCaseObj.PC_Address__c = newCaseObj.R_Address__c;
      newCaseObj.PC_State__c = newCaseObj.R_State__c;
      newCaseObj.PC_Postcode__c = newCaseObj.R_Postcode__c;
      newCaseObj.Relationship_to_child__c = newCaseObj.Referral_Relationship_to_Child__c;
      newCaseObj.ReferalContentDocumetId__c = newCaseObj.ParentContentDocumentId__c;
    }

    if (savedScreen.size() > 0)
      newCaseObj.Saved_Screen__c = String.join(
        new List<String>(savedScreen),
        ','
      );

    upsert newCaseObj;

    upsert newApplicationParticipantList;

    if (
      formCompleted != null &&
      formCompleted != '' &&
      String.isNotEmpty(formCompleted) &&
      String.isNotBlank(formCompleted) &&
      formCompleted == '100'
    ) {
      String emailId = newCaseObj.R_Email__c;
      SendEmailClass.sendEmailMethod1(
        'Application Form Completed',
        caseList[0],
        emailId
      );
      if (
        (socialWorkerContactObj != null &&
        socialWorkerContactObj != contactObj &&
        socialWorkerContactObj.Id != null &&
        socialWorkerContactObj.Email != null &&
        socialWorkerContactObj.Email != '' &&
        String.isNotEmpty(socialWorkerContactObj.Email) &&
        String.isNotBlank(socialWorkerContactObj.Email)) ||
        (doctorContactObj != null &&
```

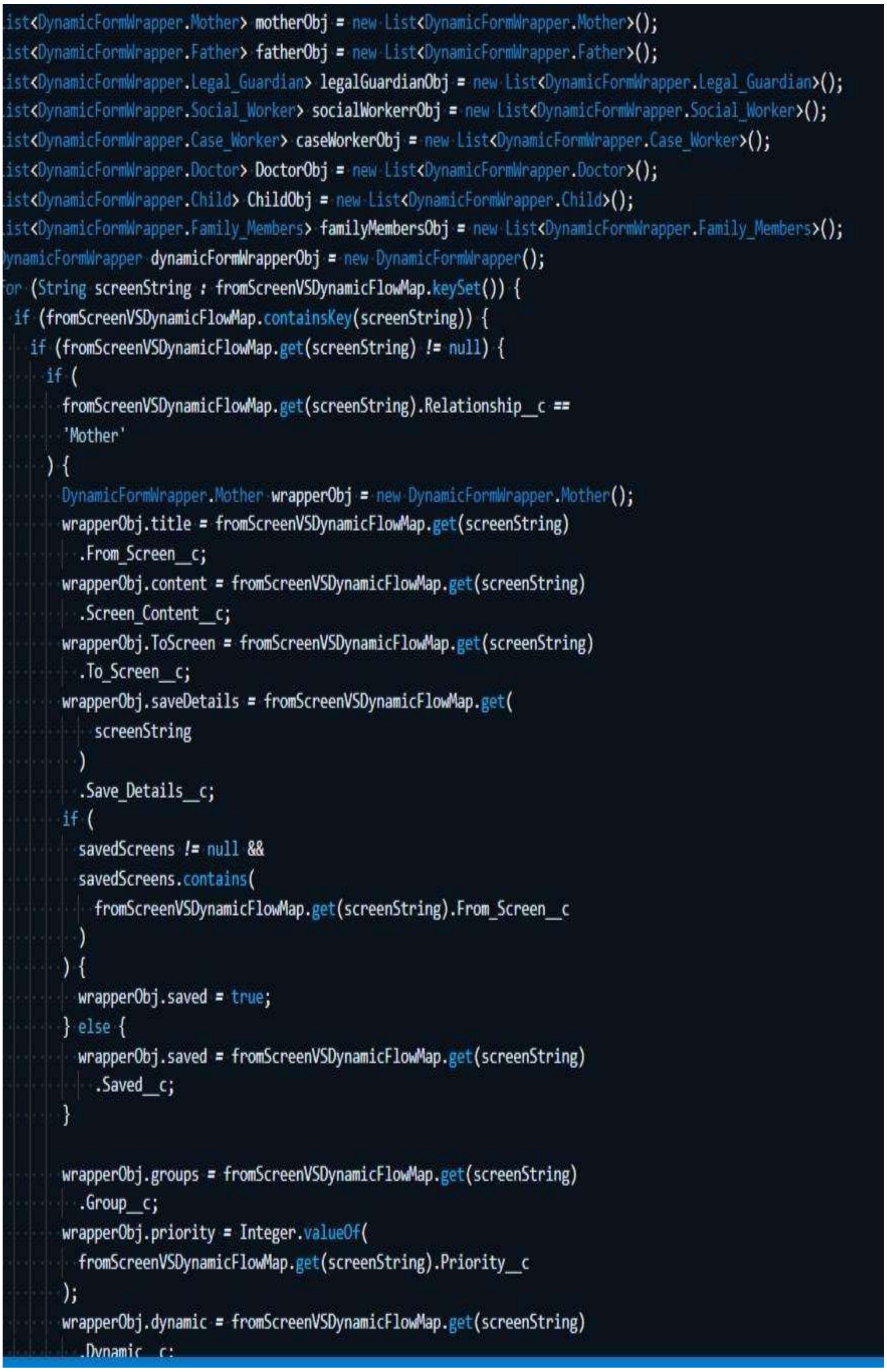


main > default > classes > DynamicFormRest.cls

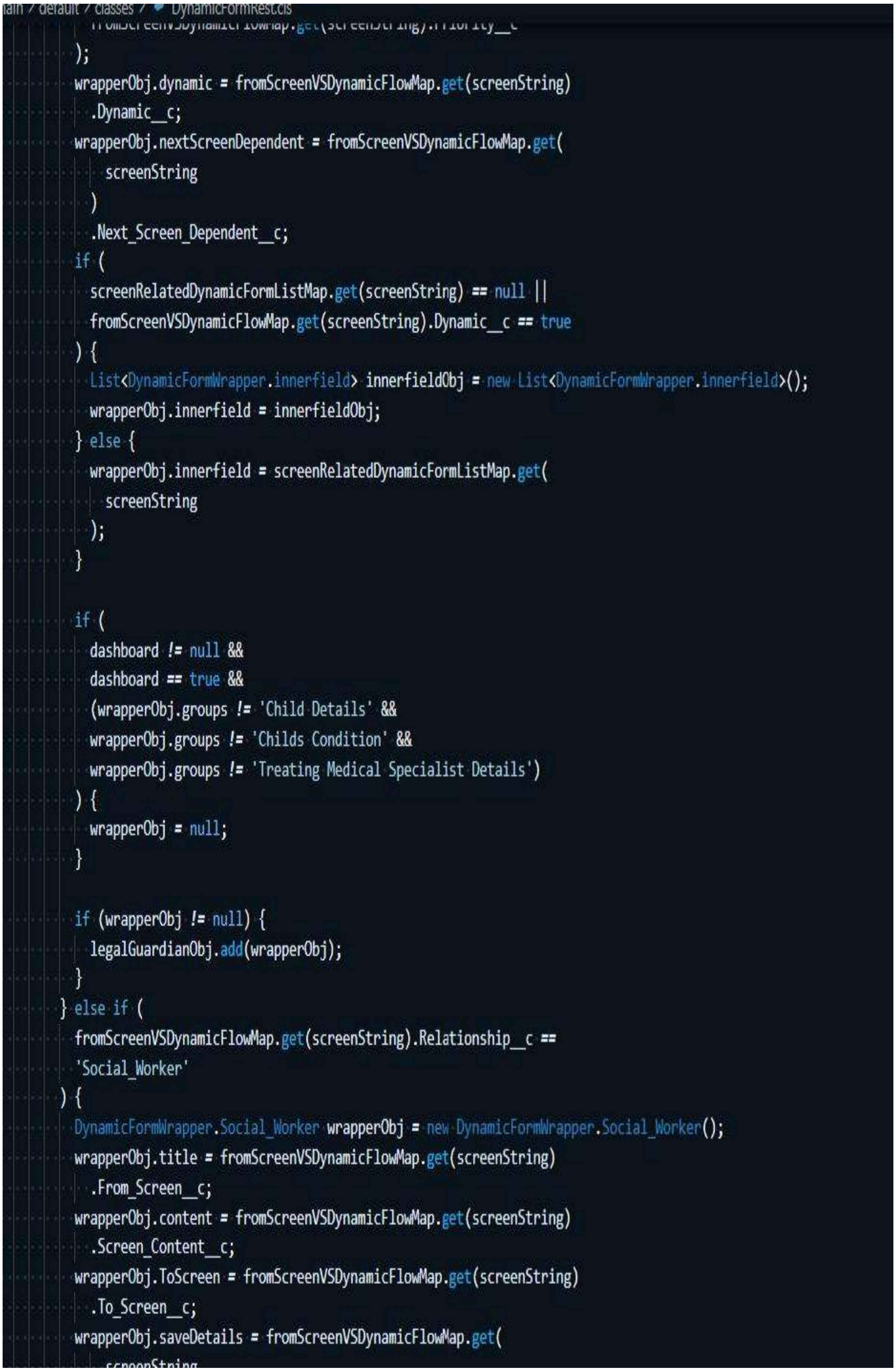
```
) {
    List<DynamicFormWrapper.Mother> motherObj = new List<DynamicFormWrapper.Mother>();
    List<DynamicFormWrapper.Father> fatherObj = new List<DynamicFormWrapper.Father>();
    List<DynamicFormWrapper.Legal_Guardian> legalGuardianObj = new List<DynamicFormWrapper.Legal_Guardian>();
    List<DynamicFormWrapper.Social_Worker> socialWorkerrObj = new List<DynamicFormWrapper.Social_Worker>();
    List<DynamicFormWrapper.Case_Worker> caseWorkerObj = new List<DynamicFormWrapper.Case_Worker>();
    List<DynamicFormWrapper.Doctor> DoctorObj = new List<DynamicFormWrapper.Doctor>();
    List<DynamicFormWrapper.Child> ChildObj = new List<DynamicFormWrapper.Child>();
    List<DynamicFormWrapper.Family_Members> familyMembersObj = new List<DynamicFormWrapper.Family_Members>();
    DynamicFormWrapper dynamicFormWrapperObj = new DynamicFormWrapper();
    for (String screenString : fromScreenVSDynamicFlowMap.keySet()) {
      if (fromScreenVSDynamicFlowMap.containsKey(screenString)) {
        if (fromScreenVSDynamicFlowMap.get(screenString) != null) {
          if (
            fromScreenVSDynamicFlowMap.get(screenString).Relationship__c ==
            'Mother'
          ) {
            DynamicFormWrapper.Mother wrapperObj = new DynamicFormWrapper.Mother();
            wrapperObj.title = fromScreenVSDynamicFlowMap.get(screenString)
              .From_Screen__c;
            wrapperObj.content = fromScreenVSDynamicFlowMap.get(screenString)
              .Screen_Content__c;
            wrapperObj.ToScreen = fromScreenVSDynamicFlowMap.get(screenString)
              .To_Screen__c;
            wrapperObj.saveDetails = fromScreenVSDynamicFlowMap.get(
                screenString
              )
              .Save_Details__c;
            if (
              savedScreens != null &&
              savedScreens.contains(
                fromScreenVSDynamicFlowMap.get(screenString).From_Screen__c
              )
            ) {
              wrapperObj.saved = true;
            } else {
              wrapperObj.saved = fromScreenVSDynamicFlowMap.get(screenString)
                .Saved__c;
            }

            wrapperObj.groups = fromScreenVSDynamicFlowMap.get(screenString)
              .Group__c;
            wrapperObj.priority = Integer.valueOf(
              fromScreenVSDynamicFlowMap.get(screenString).Priority__c
            );
            wrapperObj.dynamic = fromScreenVSDynamicFlowMap.get(screenString)
              .Dynamic__c;
```



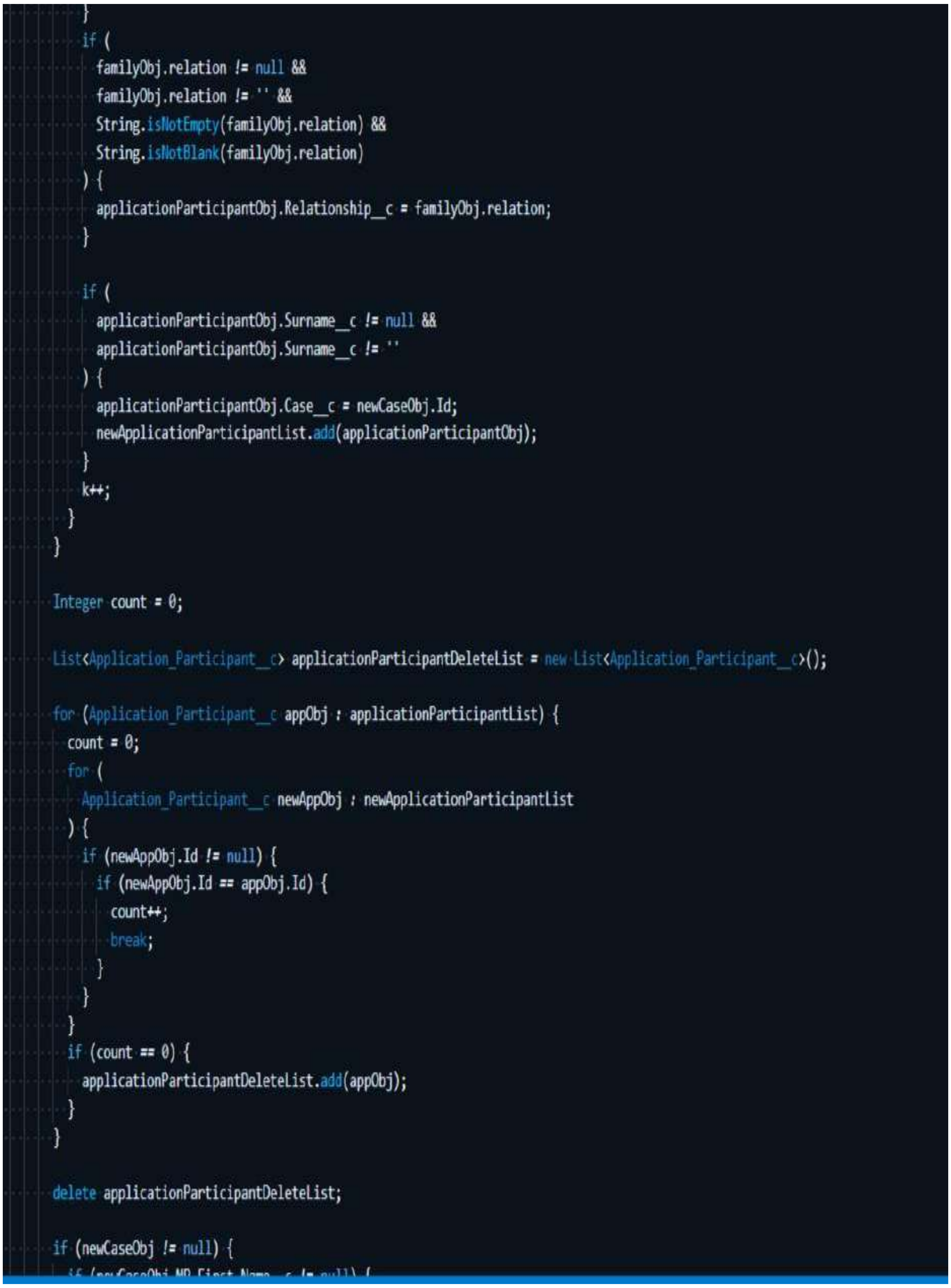
```
} else {
  caseQuery =
    'SELECT ' +
    caseFields +
    ', Specification_Related_Medical_Condition__c, Specification_Related_Wish__c, Fit_For_Wish__c, Child_Condition_by_Doctor__c FRO
    caseList[0].Id +
    '\' ORDER BY Id ASC LIMIT 1';
}

if (applicationParticipantList.size() > 0) {
  familyQuery =
    'SELECT ' +
    applicationParticipantFields +
    ' FROM Application_Participant__c WHERE Id IN: appParticipantIdSet LIMIT 5000';
}

caseResult = Database.query(caseQuery);
if (familyQuery != null) {
  familyResult = Database.query(familyQuery);
}

Integer i = 0;
for (Dynamic_Form__mdt dynamicFormObj : dynamicFormList) {
  if (
    !screenRelatedDynamicFormListMap.containsKey(
      dynamicFormObj.Relationship__c + dynamicFormObj.Screen__c
    )
  ) {
    screenRelatedDynamicFormListMap.put(
      dynamicFormObj.Relationship__c + dynamicFormObj.Screen__c,
      new List<DynamicFormWrapper.innerfield>()
    );
  }
  DynamicFormWrapper.innerfield innerfieldObj = new DynamicFormWrapper.innerfield();
  if (dynamicFormObj.Class__c == null) {
    innerfieldObj.cssClass = '';
  } else {
    innerfieldObj.cssClass = dynamicFormObj.Class__c;
  }
  innerfieldObj.disabled = dynamicFormObj.Disabled__c;
  innerfieldObj.field = dynamicFormObj.Field__c;
  if (dynamicFormObj.Label__c == null) {
    innerfieldObj.label = '';
  } else {
    innerfieldObj.label = dynamicFormObj.Label__c;
  }
```



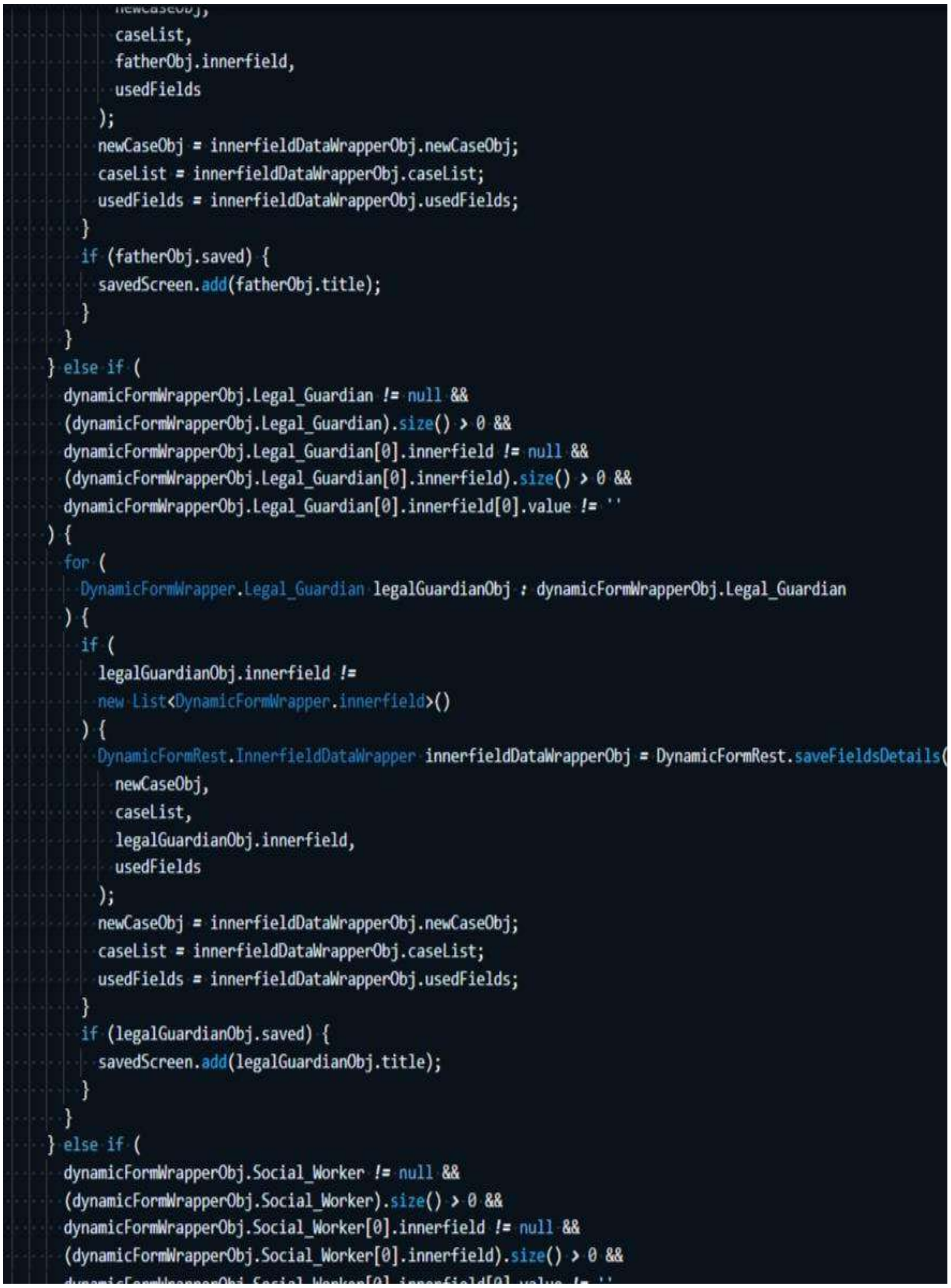
```
ist<DynamicFormWrapper.Mother> motherObj = new List<DynamicFormWrapper.Mother>();
ist<DynamicFormWrapper.Father> fatherObj = new List<DynamicFormWrapper.Father>();
ist<DynamicFormWrapper.Legal_Guardian> legalGuardianObj = new List<DynamicFormWrapper.Legal_Guardian>();
ist<DynamicFormWrapper.Social_Worker> socialWorkerrObj = new List<DynamicFormWrapper.Social_Worker>();
ist<DynamicFormWrapper.Case_Worker> caseWorkerObj = new List<DynamicFormWrapper.Case_Worker>();
ist<DynamicFormWrapper.Doctor> DoctorObj = new List<DynamicFormWrapper.Doctor>();
ist<DynamicFormWrapper.Child> ChildObj = new List<DynamicFormWrapper.Child>();
ist<DynamicFormWrapper.Family_Members> familyMembersObj = new List<DynamicFormWrapper.Family_Members>();
ynamicFormWrapper dynamicFormWrapperObj = new DynamicFormWrapper();
or (String screenString : fromScreenVSDynamicFlowMap.keySet()) {
 if (fromScreenVSDynamicFlowMap.containsKey(screenString)) {
   if (fromScreenVSDynamicFlowMap.get(screenString) != null) {
     if (
       fromScreenVSDynamicFlowMap.get(screenString).Relationship__c ==
       'Mother'
     ) {
       DynamicFormWrapper.Mother wrapperObj = new DynamicFormWrapper.Mother();
       wrapperObj.title = fromScreenVSDynamicFlowMap.get(screenString)
         .From_Screen__c;
       wrapperObj.content = fromScreenVSDynamicFlowMap.get(screenString)
         .Screen_Content__c;
       wrapperObj.ToScreen = fromScreenVSDynamicFlowMap.get(screenString)
         .To_Screen__c;
       wrapperObj.saveDetails = fromScreenVSDynamicFlowMap.get(
           screenString
         )
         .Save_Details__c;
       if (
         savedScreens != null &&
         savedScreens.contains(
           fromScreenVSDynamicFlowMap.get(screenString).From_Screen__c
         )
       ) {
         wrapperObj.saved = true;
       } else {
         wrapperObj.saved = fromScreenVSDynamicFlowMap.get(screenString)
           .Saved__c;
       }

       wrapperObj.groups = fromScreenVSDynamicFlowMap.get(screenString)
         .Group__c;
       wrapperObj.priority = Integer.valueOf(
         fromScreenVSDynamicFlowMap.get(screenString).Priority__c
       );
       wrapperObj.dynamic = fromScreenVSDynamicFlowMap.get(screenString)
         .Dynamic__c;
```



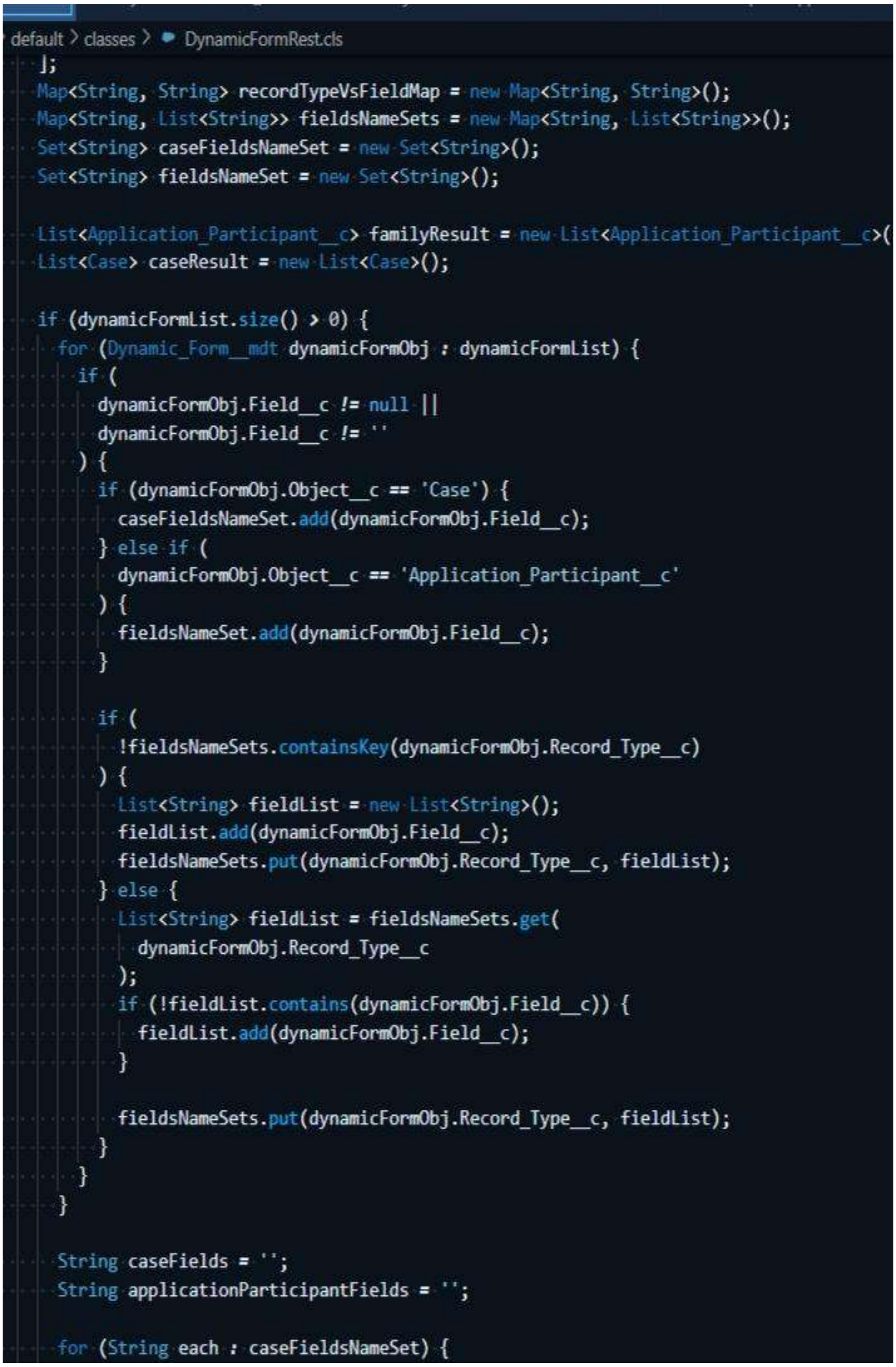
```
ain / default / classes / DynamicFormRest.cls
      fromScreenVSDynamicFlowMap.get(screenString).Priority__c
  );
  wrapperObj.dynamic = fromScreenVSDynamicFlowMap.get(screenString)
    .Dynamic__c;
  wrapperObj.nextScreenDependent = fromScreenVSDynamicFlowMap.get(
      screenString
    )
    .Next_Screen_Dependent__c;
  if (
    screenRelatedDynamicFormListMap.get(screenString) == null ||
    fromScreenVSDynamicFlowMap.get(screenString).Dynamic__c == true
  ) {
    List<DynamicFormWrapper.innerfield> innerfieldObj = new List<DynamicFormWrapper.innerfield>();
    wrapperObj.innerfield = innerfieldObj;
  } else {
    wrapperObj.innerfield = screenRelatedDynamicFormListMap.get(
      screenString
    );
  }

  if (
    dashboard != null &&
    dashboard == true &&
    (wrapperObj.groups != 'Child Details' &&
    wrapperObj.groups != 'Childs Condition' &&
    wrapperObj.groups != 'Treating Medical Specialist Details')
  ) {
    wrapperObj = null;
  }

  if (wrapperObj != null) {
    legalGuardianObj.add(wrapperObj);
  }
} else if (
  fromScreenVSDynamicFlowMap.get(screenString).Relationship__c ==
  'Social_Worker'
) {
  DynamicFormWrapper.Social_Worker wrapperObj = new DynamicFormWrapper.Social_Worker();
  wrapperObj.title = fromScreenVSDynamicFlowMap.get(screenString)
    .From_Screen__c;
  wrapperObj.content = fromScreenVSDynamicFlowMap.get(screenString)
    .Screen_Content__c;
  wrapperObj.ToScreen = fromScreenVSDynamicFlowMap.get(screenString)
    .To_Screen__c;
  wrapperObj.saveDetails = fromScreenVSDynamicFlowMap.get(
      screenString
```



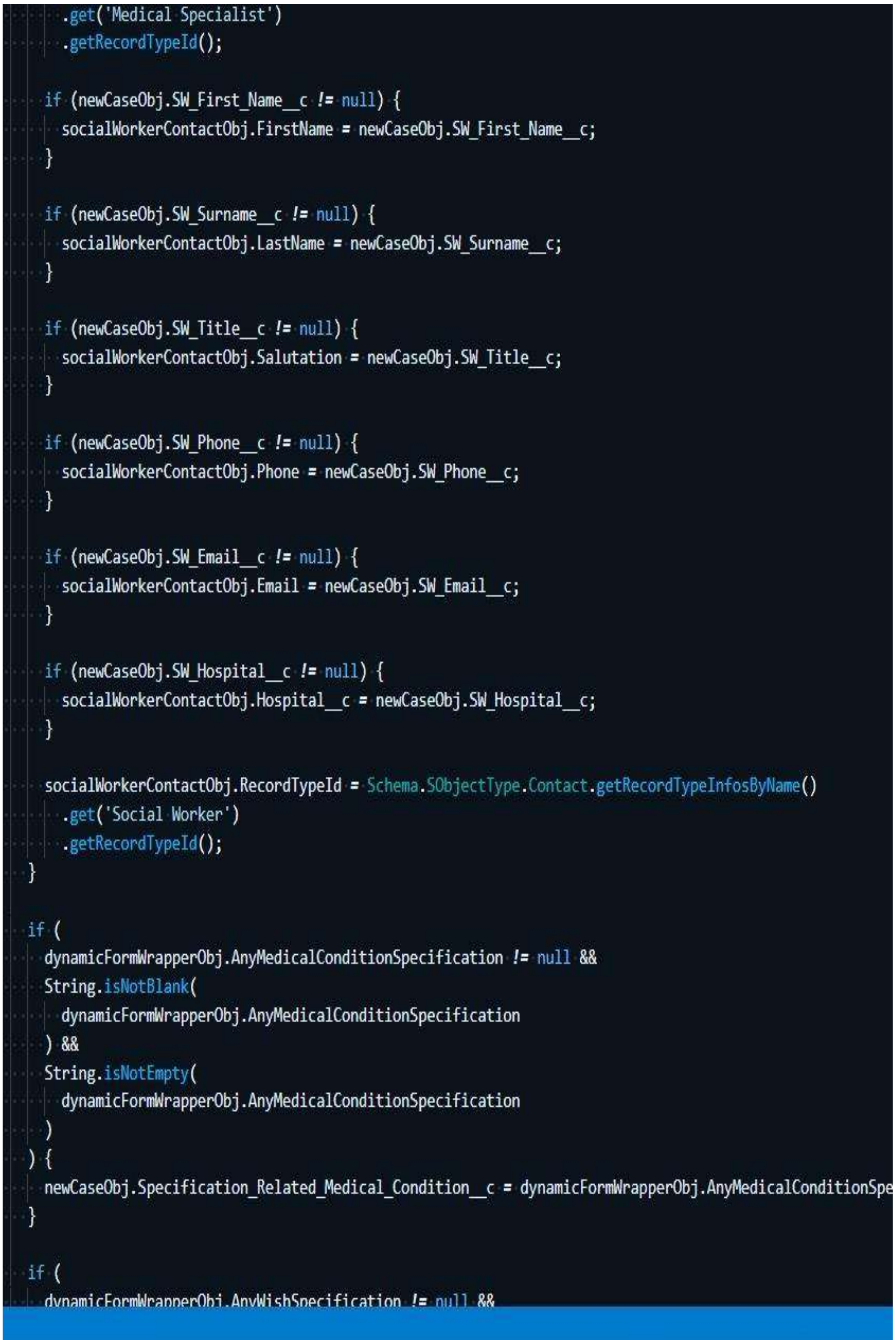
```
        }
        if (
          familyObj.relation != null &&
          familyObj.relation != '' &&
          String.isNotEmpty(familyObj.relation) &&
          String.isNotBlank(familyObj.relation)
        ) {
          applicationParticipantObj.Relationship__c = familyObj.relation;
        }

        if (
          applicationParticipantObj.Surname__c != null &&
          applicationParticipantObj.Surname__c != ''
        ) {
          applicationParticipantObj.Case__c = newCaseObj.Id;
          newApplicationParticipantList.add(applicationParticipantObj);
        }
        k++;
      }
    }

    Integer count = 0;

    List<Application_Participant__c> applicationParticipantDeleteList = new List<Application_Participant__c>();

    for (Application_Participant__c appObj : applicationParticipantList) {
      count = 0;
      for (
        Application_Participant__c newAppObj : newApplicationParticipantList
      ) {
        if (newAppObj.Id != null) {
          if (newAppObj.Id == appObj.Id) {
            count++;
            break;
          }
        }
      }
      if (count == 0) {
        applicationParticipantDeleteList.add(appObj);
      }
    }

    delete applicationParticipantDeleteList;

    if (newCaseObj != null) {
```



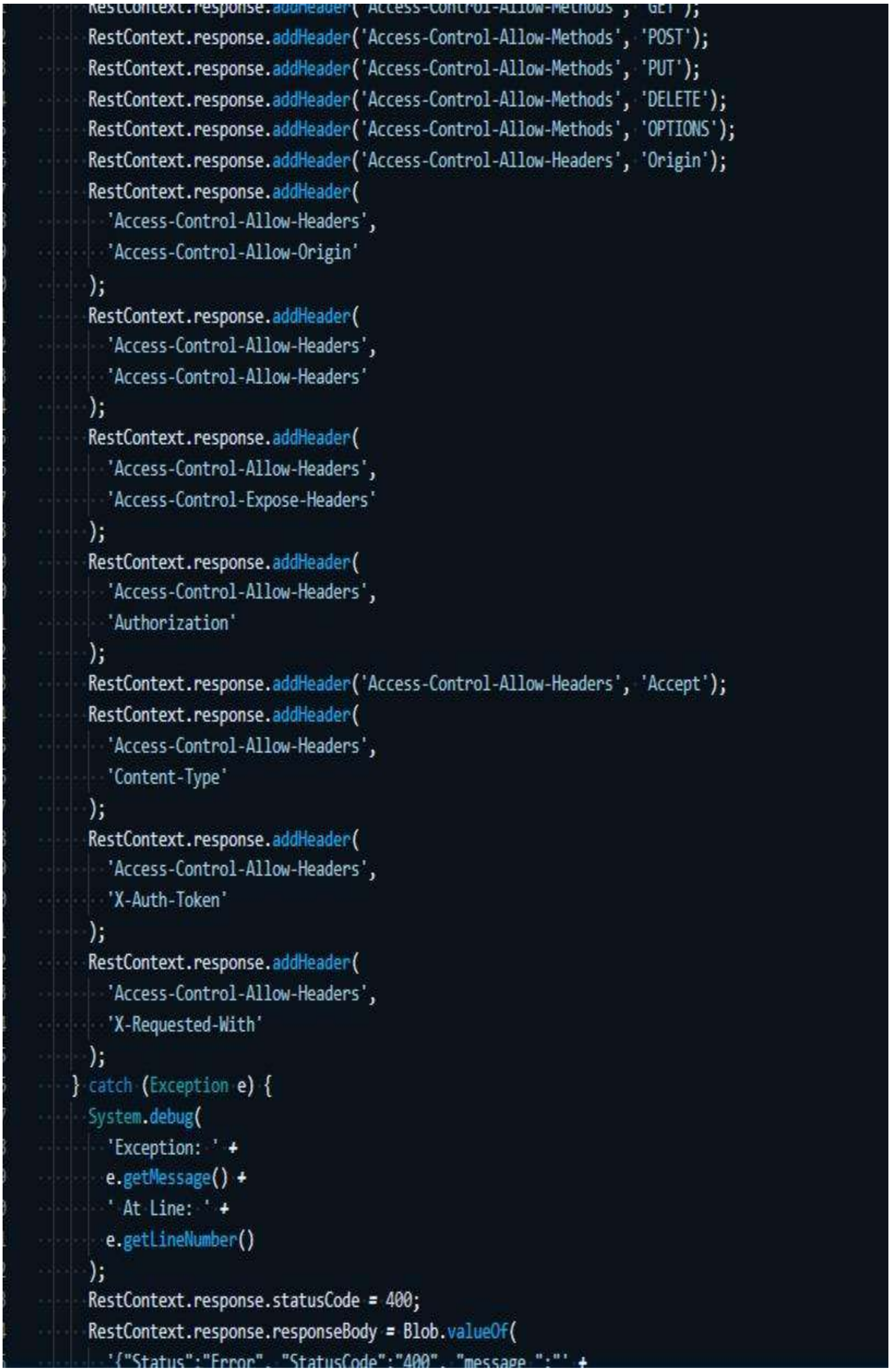
```
                newCaseObj,
                caseList,
                fatherObj.innerfield,
                usedFields
            );
            newCaseObj = innerfieldDataWrapperObj.newCaseObj;
            caseList = innerfieldDataWrapperObj.caseList;
            usedFields = innerfieldDataWrapperObj.usedFields;
        }
        if (fatherObj.saved) {
            savedScreen.add(fatherObj.title);
        }
    }
} else if (
    dynamicFormWrapperObj.Legal_Guardian != null &&
    (dynamicFormWrapperObj.Legal_Guardian).size() > 0 &&
    dynamicFormWrapperObj.Legal_Guardian[0].innerfield != null &&
    (dynamicFormWrapperObj.Legal_Guardian[0].innerfield).size() > 0 &&
    dynamicFormWrapperObj.Legal_Guardian[0].innerfield[0].value != ''
) {
    for (
        DynamicFormWrapper.Legal_Guardian legalGuardianObj : dynamicFormWrapperObj.Legal_Guardian
    ) {
        if (
            legalGuardianObj.innerfield !=
            new List<DynamicFormWrapper.innerfield>()
        ) {
            DynamicFormRest.InnerfieldDataWrapper innerfieldDataWrapperObj = DynamicFormRest.saveFieldsDetails(
                newCaseObj,
                caseList,
                legalGuardianObj.innerfield,
                usedFields
            );
            newCaseObj = innerfieldDataWrapperObj.newCaseObj;
            caseList = innerfieldDataWrapperObj.caseList;
            usedFields = innerfieldDataWrapperObj.usedFields;
        }
        if (legalGuardianObj.saved) {
            savedScreen.add(legalGuardianObj.title);
        }
    }
} else if (
    dynamicFormWrapperObj.Social_Worker != null &&
    (dynamicFormWrapperObj.Social_Worker).size() > 0 &&
    dynamicFormWrapperObj.Social_Worker[0].innerfield != null &&
    (dynamicFormWrapperObj.Social_Worker[0].innerfield).size() > 0 &&
    dynamicFormWrapperObj.Social_Worker[0].innerfield[0].value != ''
```



```
    ];
    Map<String, String> recordTypeVsFieldMap = new Map<String, String>();
    Map<String, List<String>> fieldsNameSets = new Map<String, List<String>>();
    Set<String> caseFieldsNameSet = new Set<String>();
    Set<String> fieldsNameSet = new Set<String>();

    List<Application_Participant__c> familyResult = new List<Application_Participant__c>(
    List<Case> caseResult = new List<Case>();

    if (dynamicFormList.size() > 0) {
        for (Dynamic_Form__mdt dynamicFormObj : dynamicFormList) {
            if (
                dynamicFormObj.Field__c != null ||
                dynamicFormObj.Field__c != ''
            ) {
                if (dynamicFormObj.Object__c == 'Case') {
                    caseFieldsNameSet.add(dynamicFormObj.Field__c);
                } else if (
                    dynamicFormObj.Object__c == 'Application_Participant__c'
                ) {
                    fieldsNameSet.add(dynamicFormObj.Field__c);
                }

                if (
                    !fieldsNameSets.containsKey(dynamicFormObj.Record_Type__c)
                ) {
                    List<String> fieldList = new List<String>();
                    fieldList.add(dynamicFormObj.Field__c);
                    fieldsNameSets.put(dynamicFormObj.Record_Type__c, fieldList);
                } else {
                    List<String> fieldList = fieldsNameSets.get(
                        dynamicFormObj.Record_Type__c
                    );
                    if (!fieldList.contains(dynamicFormObj.Field__c)) {
                        fieldList.add(dynamicFormObj.Field__c);
                    }

                    fieldsNameSets.put(dynamicFormObj.Record_Type__c, fieldList);
                }
            }
        }

        String caseFields = '';
        String applicationParticipantFields = '';

        for (String each : caseFieldsNameSet) {
```



```
        .get('Medical Specialist')
        .getRecordTypeId();

    if (newCaseObj.SW_First_Name__c != null) {
      socialWorkerContactObj.FirstName = newCaseObj.SW_First_Name__c;
    }

    if (newCaseObj.SW_Surname__c != null) {
      socialWorkerContactObj.LastName = newCaseObj.SW_Surname__c;
    }

    if (newCaseObj.SW_Title__c != null) {
      socialWorkerContactObj.Salutation = newCaseObj.SW_Title__c;
    }

    if (newCaseObj.SW_Phone__c != null) {
      socialWorkerContactObj.Phone = newCaseObj.SW_Phone__c;
    }

    if (newCaseObj.SW_Email__c != null) {
      socialWorkerContactObj.Email = newCaseObj.SW_Email__c;
    }

    if (newCaseObj.SW_Hospital__c != null) {
      socialWorkerContactObj.Hospital__c = newCaseObj.SW_Hospital__c;
    }

    socialWorkerContactObj.RecordTypeId = Schema.SObjectType.Contact.getRecordTypeInfosByName()
      .get('Social Worker')
      .getRecordTypeId();
  }

  if (
    dynamicFormWrapperObj.AnyMedicalConditionSpecification != null &&
    String.isNotBlank(
      dynamicFormWrapperObj.AnyMedicalConditionSpecification
    ) &&
    String.isNotEmpty(
      dynamicFormWrapperObj.AnyMedicalConditionSpecification
    )
  ) {
    newCaseObj.Specification_Related_Medical_Condition__c = dynamicFormWrapperObj.AnyMedicalConditionSpe
  }

  if (
    dynamicFormWrapperObj.AnyWishSpecification != null &&
```



```
    RestContext.response.addHeader('Access-Control-Allow-Methods', 'GET');
    RestContext.response.addHeader('Access-Control-Allow-Methods', 'POST');
    RestContext.response.addHeader('Access-Control-Allow-Methods', 'PUT');
    RestContext.response.addHeader('Access-Control-Allow-Methods', 'DELETE');
    RestContext.response.addHeader('Access-Control-Allow-Methods', 'OPTIONS');
    RestContext.response.addHeader('Access-Control-Allow-Headers', 'Origin');
    RestContext.response.addHeader(
      'Access-Control-Allow-Headers',
      'Access-Control-Allow-Origin'
    );
    RestContext.response.addHeader(
      'Access-Control-Allow-Headers',
      'Access-Control-Allow-Headers'
    );
    RestContext.response.addHeader(
      'Access-Control-Allow-Headers',
      'Access-Control-Expose-Headers'
    );
    RestContext.response.addHeader(
      'Access-Control-Allow-Headers',
      'Authorization'
    );
    RestContext.response.addHeader('Access-Control-Allow-Headers', 'Accept');
    RestContext.response.addHeader(
      'Access-Control-Allow-Headers',
      'Content-Type'
    );
    RestContext.response.addHeader(
      'Access-Control-Allow-Headers',
      'X-Auth-Token'
    );
    RestContext.response.addHeader(
      'Access-Control-Allow-Headers',
      'X-Requested-With'
    );
  } catch (Exception e) {
    System.debug(
      'Exception: ' +
      e.getMessage() +
      ' At Line: ' +
      e.getLineNumber()
    );
    RestContext.response.statusCode = 400;
    RestContext.response.responseBody = Blob.valueOf(
      '{"Status":"Error", "StatusCode":"400", "message ":"' +
```

Chapter – 4

Testing

The Force.com platform requires that at least 75% of the Apex Code in an org be executed via unit tests in order to deploy the code to production. You shouldn’t consider 75% code coverage to be an end goal though. Instead, you should strive to increase the state coverage of your unit tests. Code has many more possible states than it has lines of code. For example, the following method has 4,294,967,296 different states: System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic.

4.1 TYPES OF TESTING

BLACK BOX TESTING:

The technique of testing without having any knowledge of the interior workings of the application is called black box testing. The tester is oblivious to the system architecture and does not have access to the source code. Typically, while performing a black box test, a tester will interact with the system's user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon.

WHITE BOX TESTING:

Whitebox testing is the detailed investigation of internal logic and structure of the code. Whitebox testing is also called glass testing or open box testing. In order to perform white box testing on an application, a tester needs to know the internal workings of the code.

GREY BOX TESTING:

Grey box testing is a technique to test the application with having a limited knowledge of the internal workings of an application. In software testing, the phrase the more you know, the better carries a lot of weight while testing an application.

UNIT TESTING:

Unit Testing contains the testing of each unit of Recruitment Application. We have tested each interface by input values and check whether it is working properly working or not we also tested database connectivity. We have entered value in interface and check that the values are properly goes to corresponding tuples or not.

INTEGRATION TESTING:

Integration testing is defined as the testing of combined parts of an application to determine if they function correctly. Integration testing can be done in two ways: Bottom up integration testing and Top down integration testing.

SYSTEM TESTING:

System testing tests the system as a whole. Once all the components are integrated, the application as a whole is tested rigorously to see that it meets the specified Quality Standards. This type of testing is performed by a specialized testing team

AUTOMATION T:

Automation Testing or Test Automation is a software testing technique that performs using special automated testing software tools to execute a test case suite. On the contrary, Manual Testing is performed by a human sitting in front of a computer carefully executing the test steps.[15]

CHAPTER 5

CONCLUSION

5.1 CONCLUSION

The main agenda of this project was to meet client needs with in the given period of time. In this project, I have worked on different parts of Salesforce including Lightning Components, VisualForce Pages, Lightning Web Components, Apex classes, triggers etc. Through all these tasks I have learned a lot of things about salesforce and its implementation and gained knowledge and skills. Integration with a third party app i.e. Twilio was the most interesting part for me as it was totally new thing for me and was working first time on integration and Webservices. In this whole process I have used

- Apex Language
- Java Script
- HTML
- XML
- Twilio (For integration)
- VisualForce Page
- Triggers
- Webservices

Chapter – 6

Bibliography

- trailhead.salesforce.com
- developer.salesforce.com
- https://www.lightningdesignsystem.com/
- https://developer.salesforce.com/docs/component-library/overview/components
- Communityforce.com

REFERENCES

[1] Gregory S. Smith,“Cloud Computing Strategies and Risks”, 11 April 2013, Chapter 7, Pages 125-153, Print ISBN:9781118390030

[2] Davis D. Janowski,“Selecting the Right CRM System”, January 2013, Pages 5-15, Print ISBN:9781118434765

[3] Mehmet N. Aydin,NazimZiya, “Cloud-Based Development Environments”, Pages 62-69, 13 May 2016, Print ISBN:9781118821978

[4] Tansu Barker, “Benchmarks of Successful Salesforce Performance”, 08 April 2009

[5] DevenN.Shah, Dilip Motwani, “Software Engineering”, Publisher: Dreamtech Press,2010, ISBN:9350040395, 9789350040393

[6] Gerard O’Regan, “Concise Guide to Software Engineering From Fundamentals to Application Methods”,2017, Publisher: Springer International Publishing, ISBN: 978-3-319-57750-0

[7] IztokFajfar, “Start Programming Using HTML, CSS And JavaScript”, 2015, ISBN: 9780429083457, Imprint: Chapman and Hall/CRC

[8] Rajesh K.Maurya and Swati R. Maurya, “Software Testing”, April 21, 2021, Publisher: Dreamtech Press, ISBN: 978-9350044001

[9] Rod Stephens, “Beginning Software Engineering”, Publisher: Wrox, Publication date: 24 April 2015, ISBN: 8126555378

[10] Chester Bullock, Mark Pollard, “salesforce-marketing-cloud-for- dummies”, Publisher: For Dummies,22 December 2017, ISBN: 1119122090

[11] Rod Stephens, “Beginning Software Engineering”, Publisher: Wrox,2 March 2015, ISBN: 978-1118969144

[12] SaeidAbolfazli, ZohrehSanaei, Mohammad HadiSanaei, Mohammad Shojafar and Abdullah Gani, “Mobile Cloud Computing”, Publisher: Wiley,13 May 2016

[13] Ernesto Exposito andCodé Diop, “Service-Oriented and Cloud Computing Architectures”, Publisher: Wiley, July 2014, ISBN: 978-1-118-76169-4

[14] Xiaolong Li, “Information Technology and Applications”, Publisher:Boca Raton - CRC Press, 2014, ISBN:9780429226373

[15] XiaoxuDiao, Manuel Rodriguez and Carol Smidts,“Automated Software Testing”, First published: 06 July 2018