

Deliverable 5 Report

Team Squad 8

A. Deliverable 5 Requirements' Status:

Requirements:

Section No.	Requirement Name	Status	Description
3.3.1	Doctor Appointment Booking System for Patients	finished	A feature that enables the patients to book appointments from their dashboard.
3.3.2	Contact us backend to trigger emails	finished	A feature that sends confirmation mails to the email account that is used for signup regarding appointment booking confirmation, appointment cancellation and signup confirmation.
3.3.3	Fixes for MedVoyage favicon	finished	Added a cute favicon for the web application and updated logo to have transparent background
3.3.4	Help link to user manual	finished	A help link was added to the user manual that will direct the user about how to navigate through the webpages in the application
3.3.5	Book an appointment button quick fix	finished	A button was added to the patient's dashboard to book an appointment.
3.3.6	Password validation fixes	finished	Added requirements for the characters in password for extra validation and security purpose.
3.3.7	Frontend logic for booking	finished	Implemented logic to show dropdowns, available slots , doctors for booking an appointment.
3.3.8	Frontend fixes for reset password page	finished	Implemented few fixes in the frontend part for the rest password page.
3.3.9	Frontend fixes for signup	finished	Implemented few fixes in the frontend part for the signup page.
3.3.10	Added unavailability for booked slots	finished	Added a feature that adds an unavailable flag to the already booked slots and doesn't fetch them to the dropdown.
3.3.11	Bug fixes on all pages	finished	HTML title fixes on all pages

Deliverable 5 Report

Team Squad 8

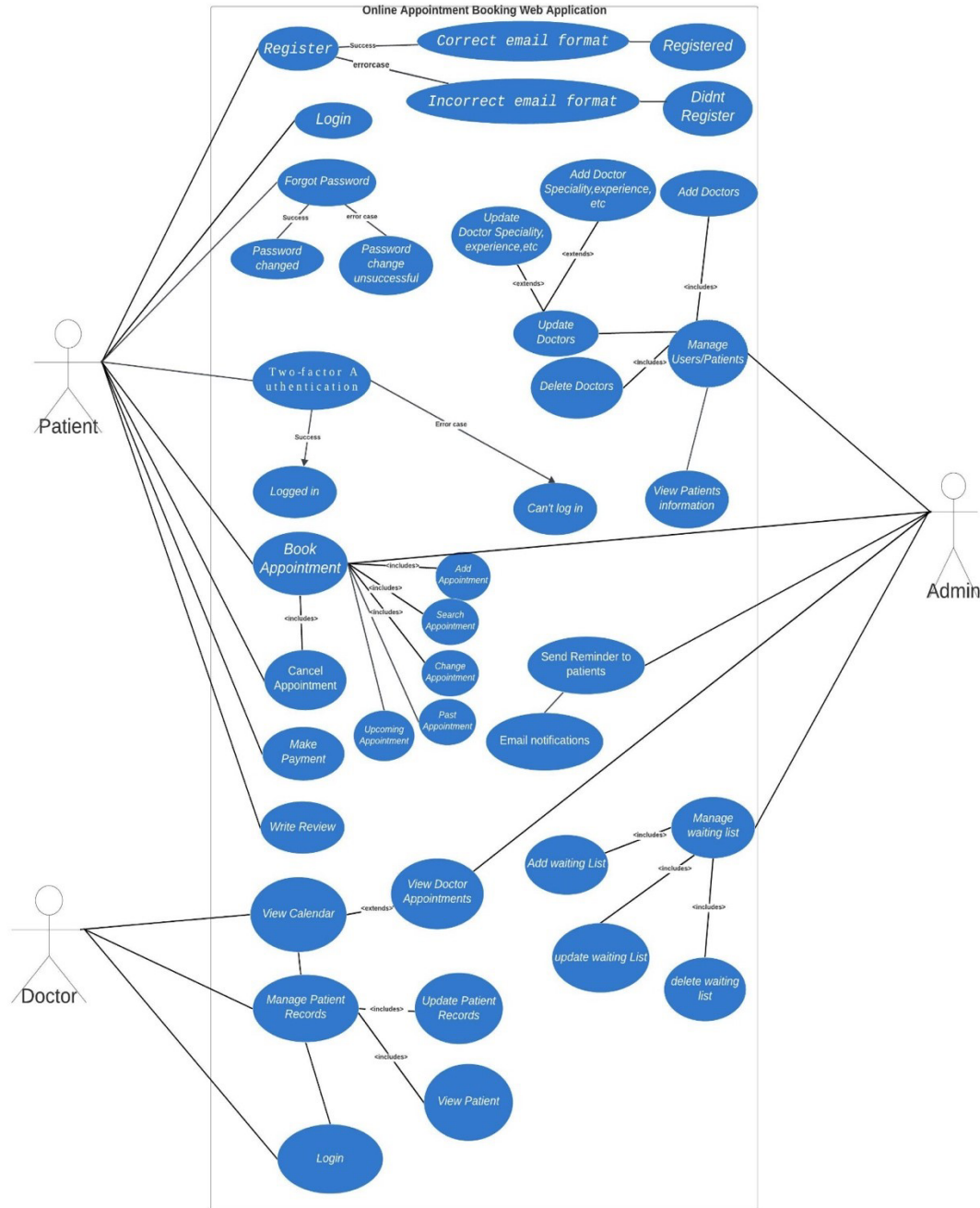
3.3.12	Fetch profile pics on about us	finished	A feature that fetches the profile picture from GitHub to display on the about us page.
3.3.13	Contact us details	finished	Updated the contact us details on the contact us page
3.3.14	Color fixes	finished	Color fixes on all pages to have MedVoyage theme green.
3.3.15	Fetch doctor details	finished	A feature that fetches the names of the doctors into the dropdown while patient books an appointment.
3.3.16	Production bug	Finished	Done some fixes to the settings file for CSRF and static file locations
3.3.17	Adding Docker file	finished	Added Docker file to make Django app docker buildable and deployable
3.3.18	Deployment on docker container	finished	Added a full deployment script to deploy the whole webapp and MySQL server on docker container.
3.3.19	User manual	Finished	Added a help page on the navbar that consists of the user manual that guides the user about the webapp usage.
3.3.20	Backend of Appointment booking system	finished	Implemented the logic for the backend of the patient appointment booking system.
3.3.21	Phone number field validation	finished	A feature that validates 10-digit phone number format and prevents signup while signing up with an invalid phone number.
3.3.22	Login bug Fix	finished	A login redirect bug fix that makes the pages viewable only on login.
3.3.23	Patient appointments list	finished	Made changes and bug fixes to the patient's appointment list on the doctor's dashboard.
3.3.24	Appointment cancellation	finished	A feature that enables the patient to cancel an appointment that was already booked previously by them.
3.3.25	Add slots and view slots list	Finished	HTML file fixes on the add and view slots list

Deliverable 5 Report

Team Squad 8

B. UML diagrams for Deliverable 5:

a. Use Case diagram:

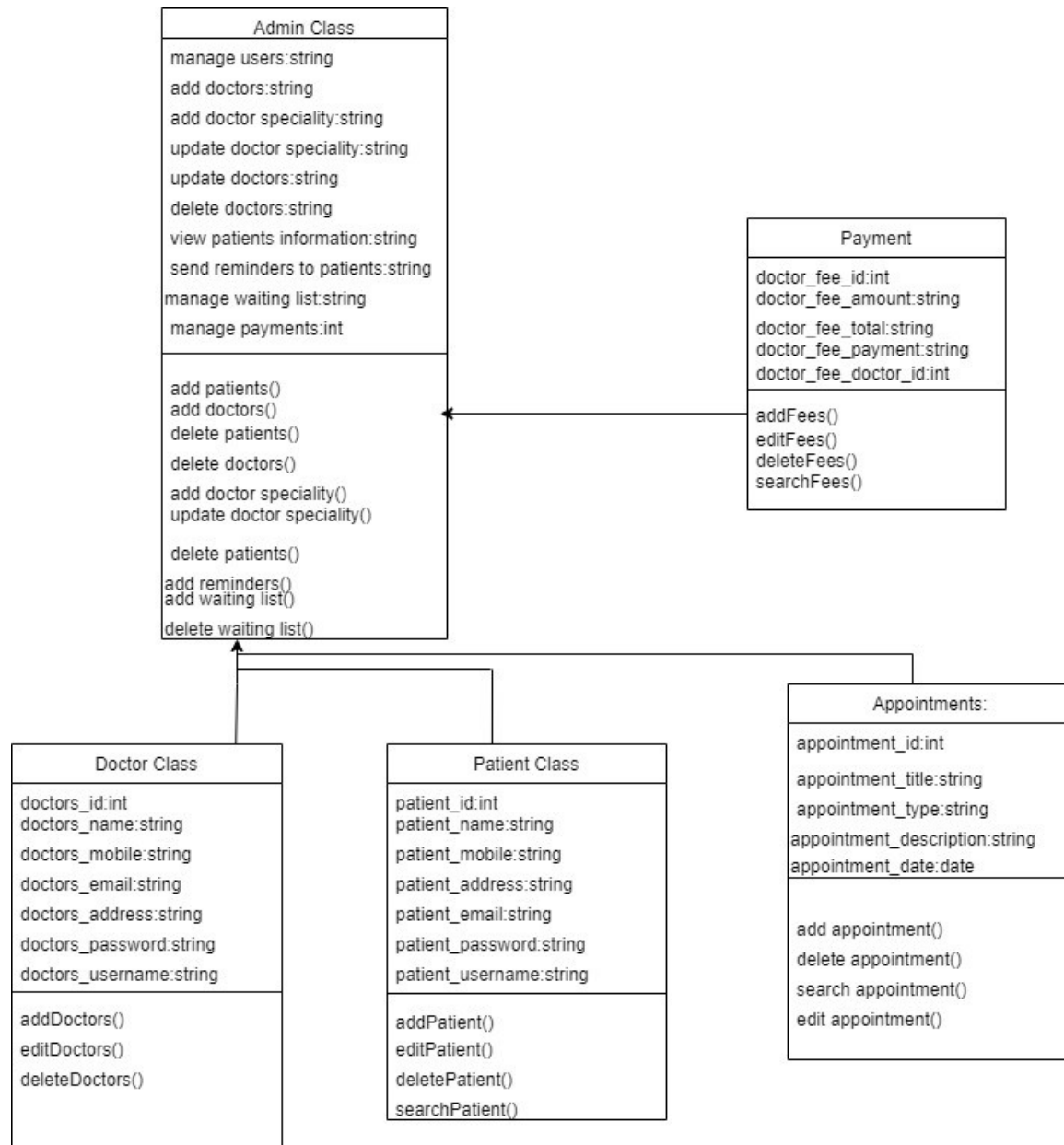


a. Use Case Diagram representing the system's functionalities and the external entities interacting with them

Deliverable 5 Report

Team Squad 8

c. Class Diagram:



c. Class Diagram showing showcasing classes, their attributes, operations, and relationship between the various classes

Deliverable 5 Report

Team Squad 8

d. Different use cases for MedVoyage application :

Use case 1:

Use Case Number	UseCase-03
Use Case Name	Registration confirmation for Patients and Doctors
Overview	The Users (Patients/Doctors) shall start entering the registration information on the MedVoyage web application
Pre-condition(s)	<p>The Users have arrived at the MedVoyage web application through the URL</p> <ul style="list-style-type: none">• User database is properly configured. A separate table for patients and a separate table for doctors• Web server is open, and the host is waiting for registrations of patients/doctors.• The new users have access to the MedVoyage web application via URL
Scenario Flow Main (success) flow:	<ol style="list-style-type: none">1.User (patient/doctor) selects the signup option instead of login.2.The MedVoyage system prompts the user(patient/doctor) for their personal information, including: Username, Password, and Email.3.The user(patient/doctor) enters the requested information.4.MedVoyage system verifies if the requested information is within the pre-specified constraints.<ol style="list-style-type: none">a.If constraints are violated, Display error messageb.Return to Step 25.User(patient/doctor) inputs the information.6.MedVoyage system verifies the information.<ol style="list-style-type: none">a.If information is invalid, display an error message specifying instructions.b.Return to Step 67.If the email is not entered in the correct format like if the @ symbol is missing or if there is .co or .com is missing then the sign up will not be possible.8.MedVoyage system displays the confirmation of signup
Alternate Flows	<ol style="list-style-type: none">1)The user confirms the MedVoyage system suggestion.2)The MedVoyage system returns to step (7).3)Two factor Authentication is applied and if it is successful then the user could login.

Deliverable 5 Report

Team Squad 8

Post Condition	New user(patient/doctor) information has been successfully added to the MYSQL database.
----------------	---

Use case 2 :

Use Case Number	UseCase-04
Use Case Name	Forgot Password
Overview	The users(doctors/patients) shall retrieve their password with forgot password option.
Pre-condition(s)	The user(doctors/patient) has already completed registration/sign up page, they have a username, but they forgot password.
Scenario Flow Main (success) flow:	<p>Main (success) flow:</p> <ol style="list-style-type: none">1.The user(doctor/patient) connects to the MedVoyage system.2.The user(doctor/patient) selects the option to login3.MedVoyage System prompts the user(doctor/patient) to enter credentials.4.The user(doctor/patient) enters credentials.5.MedVoyage System validates credential information is correct.6.The user forgot the password.7.The user clicks the forgot password button8.The username and email will be asked9.Email notification will be sent to change the password10.The user will be able to change the password
Alternate Flows	<p>At any step, the user(doctor/patient) can instead click the cancel button.</p> <p>1)Return to homepage>About Us page)</p>
Post Condition	MedVoyage System redirects users(doctors/patients) to the Login page.

Deliverable 5 Report

Team Squad 8

C. Test Cases (unit tests) for Deliverable 5:

Test Function Name	Test Description	HTTP URL	HTTP Method	Expected HTTP Response Code	Expected Outcome
test_client_dashboard_content	Tests if the fields in the navbar are rendered correctly.	/client_dashboard/	GET	200	Contains navbar fields
	Tests if the footer is displayed correctly in the dashboard.				Contains footer
	Tests whether the font and content are correct				Font and content are displayed correctly.
test_client_Profile_content	Tests if the navbar on the homepage contains all the fields, the content and edit button is displayed along with it.	/client_profile/	GET	200	Contains navbar fields
test_client_profile_rendered_correctly	Tests if the form elements are rendered correctly	/client_profile/	GET	200	Contains "© MedVoyage"
test_client_profile_labels	Tests for the labels are correctly rendered.	/client_profile/	GET	200	Contains 'MedVoyage' and the specific journey-to-health text
test_update_user_info_page_is_rendered	Tests if the navbar on the homepage contains all the fields, the content and update button is displayed along with it.	/client_profile/	GET	200	Contains: "MedVoyage About Us"
test_readonly_fields	Tests if the readonly fields are displayed as read-only fields correctly	/client_profile/			

Deliverable 5 Report

Team Squad 8

test_update_user_info_form_submission	Prepares data for form submission, checks if the form submission is successful and redirects to the expected page and checks if the user's information is updated in the database	/client_update/	POST	302	Page is successful with correct redirection to update form, updates the given user's information to the dashboard.
test__appt_nav	Checks to see if the page request with the patient appointments button is successful in the navigation and loads to view the patient appointments, once signup and log in have occurred	Various URLs ('/signup/', '/login/', '/home/')	POST GET	302 200	Page is successful with the patient appointments button loading and redirecting successfully once the specific user has logged in
test_appt_page	Checks to see if the patient appointment page with the patient's past and upcoming appointments successfully loads and redirects when the patient has signup and logged in	Various URLs ('/signup/', '/login/', '/client_appointments/')	POST GET	302 200	The patient appointment page with the patient's past and upcoming appointments successfully loads and redirects when the patient has logged in

Deliverable 5 Report

Team Squad 8

test_no_appt_listing_when_no_appt_made	Checks to see if the message “No Upcoming Appointments” displays under “Upcoming Appointments” if there are no scheduled appointments in the portal by the logged in patient	Various URLs (/signup/, /login/, /client_appointments/)	POST GET	302 200	Page successfully displays “No Upcoming Appointments” when there are no scheduled future appointments in the portal by the logged in user
test_appt_no_nav_logged_out	Checks to see if the patient appointment button doesn’t pop up in the navigation bar when no sign up and log in have occurred in terms of successful loading	/home/	GET	200	As expected, the page with the patient appointments button in the navigation bar does not pop up as the user has not logged in or signed up as expected
test_page_inaccessible_when_not_logged_in	Tests to check if the patient appointments page doesn’t display when no user has logged in	/client_appointments/	GET	200	As expected, the patient appointment page is not displayed if the user has not logged in
test_add_slot	Tests to check if doctor availability timings are recorded	/add_slot/	POST	200	Slot should be added to the database
test_delete_slot	Tests to see if an already recorded time slot is deleted	/delete_slot/	DELETE	200	Slot should be removed from the database
test_edit_slot	Tests to see if an already recorded slot is updated with changed values	/edit_slot/	PUT	200	Slot details should be updated in the database

Deliverable 5 Report

Team Squad 8

These test cases are the test cases that were written to check the functionalities of newly added features and we have also upgraded the other test cases that were previously mentioned in deliverable 4.

D. A user manual that tells us how to install/use your program. This is meant for the end-user of the software.

User Manual (more of the user manual is under “How to use the basic functionalities of the MedVoyage Web-Application” section)

The initial setup to launch our website on the browser is like the deliverable 3.

Installation of the VS Code:

Visual Studio Code (VS Code) could be used for code editing. To install it, visit the official VS Code website at <https://code.visualstudio.com/> and download the version appropriate for your operating system (Windows, macOS, or Linux).

Follow the installation instructions for your specific operating system:

- On Windows, run the installer executable and follow the on-screen instructions.
- On macOS, drag the VS Code application to the Applications folder.
- On Linux, use the provided package manager or download the .deb or .rpm package, then follow the installation instructions.

After installation, open Visual Studio Code from your applications or program menu. You can configure settings globally (user settings) or on a per-project basis (workspace settings). VS Code has built-in Git support. If you're working with a Git repository, you'll see version control icons in the left sidebar. You can stage, commit, and push changes directly from VS Code.

Docker setup:

Docker provides a consistent and isolated environment for applications, making it easier to deploy and manage software across different computing environments. To install and set up Docker, you'll need to follow specific steps based on your operating system. Docker is available for Windows, macOS, and various Linux distributions.

To download the docker desktop,

Visit the Docker Desktop for Windows page: <https://www.docker.com/products/docker-desktop>.

Visit the Docker Desktop for Mac page: <https://www.docker.com/products/docker-desktop>.

Download the Docker Desktop installer for Windows/Mac Installer. Run the installer and follow the installation wizard's instructions. During the installation, Docker may prompt you to enable Windows Subsystem for Linux (WSL) 2. Allow this, as WSL 2 is required for Docker to run, or you can install Windows subsystem for Linux using the terminal directly by running the `wsl --install` command or you could get it directly from the Microsoft store for windows OS. Once the installation is complete, Docker Desktop should be available in your windows applications. Launch it. To verify whether your Docker is configured correctly try running the `docker --version` command from your terminal.

Deliverable 5 Report

Team Squad 8

Cloning the repo into the VS Code:

If you haven't already, you'll need to install Git on your computer. You can download Git from the official website: <https://git-scm.com/downloads>.

Launch Visual Studio Code on your computer. You should be able to see clone repository in a new window. Click on clone repository and choose clone git repo, it will prompt you to add the link or repository name. Copy and paste the repo link there <https://github.com/kalyancheerla/MedVoyage>.

Choose the directory in which you want the repo to get cloned and open in a new window. This will create a new directory with all the workspace files of your repo. You can clone the repo using the terminal also. In VS Code, you can open the integrated terminal by clicking on "View" in the top menu and selecting "Terminal." Use the terminal to navigate to the directory where you want to clone the Git repository. You can use the `cd` command to change directories. Use the git clone command with the repository name as follows:

git clone <https://github.com/kalyancheerla/MedVoyage.git>

After the repository is cloned, you can open it in Visual Studio Code. You can now work on the files in the cloned repository, commit changes, and interact with Git directly from the integrated terminal or by using the source control features provided by VS Code.

Python interpreter setup:

Go to the repository cloned directory in your VS code and open a new terminal to install the Django and MySQL dependencies. In the new terminal use the following command:

```
pip install -r src/requirements.txt
```

This will open the document requirements.txt read the Django, MySQL and recursively install these dependencies.

Setup of MySQL Database in Docker:

We have already documented the tool script to run and setup the MySQL database in docker in the tools/setup_mysql_docker.sh document in our files. This script can be executed directly in Linux based environments, for windows we can execute the following commands:

```
docker run -p 3306:3306 --name mysqlinstance -e MYSQL_ROOT_PASSWORD='pa$$w0rd' -d mysql:latest
```

When you run this command, Docker will download the MySQL image if it's not already available locally, create a new container from that image, and start the MySQL database server within the container. The `-p` option maps the MySQL port so that you can connect to it from the host or another machine. The MySQL root password is set to 'pa\$\$w0rd' for authentication.

The resulting container, named "MySQL instance," will be running in the background with the MySQL server accessible on port 3306, and you can interact with it as needed. This is a common way to run a MySQL database server in a Docker container for development or testing purposes.

```
docker exec -it mysqlinstance mysql -uroot -p'pa$$w0rd'
```

Deliverable 5 Report

Team Squad 8

When one runs this command, one will be granted access to the MySQL server running inside the "mysqlinstance" container as the root user with the provided password. This allows one to interact with and manage the MySQL database server using the MySQL command-line client. One can execute SQL queries, manage databases, and perform various administrative tasks within the container's MySQL environment. MySQL command line opens, and one can create database *MedVoyagedb* and then exit from the MySQL using the exit command.

```
CREATE DATABASE MedVoyagedb;
```

```
exit
```

Creating tables and schema of models in MySQL DB:

Navigate to the directory where one has previously installed the Django and MySQL dependencies in the terminal and make migration files in that directory.

```
python src/manage.py makemigrations
```

It helps one create migration files for any changes one has made to their database schema, making it easy to apply these changes to the database when you run the "migrate" command. This is part of Django's database schema management and version control system.

```
python src/manage.py migrate
```

It helps you keep the database schema in sync with your project's models and ensures that the database is properly configured for your application. This command is typically run after creating or modifying migration files with the "makemigrations" command.

Running the Django webserver:

```
python src/manage.py runserver
```

In the same directory, after creating the migration files we can run the above command to run the webserver, and this can be viewed on localhost:8000

Running unit testcases

```
python src/manage.py test src/main/tests/
```

This is a way to execute test cases for the "main" app within your Django project. This is a crucial step in ensuring the reliability and correctness of your Django application by verifying that various components and features work as expected. Currently, we integrated GitHub Actions to run unit testcases on commits made on the main branch and pull requests that are created.

As an extension and to improve the accessibility for the users, in deliverable 5, we have managed to host our application on a free domain. You can directly access and check the progress of our web application at <https://medvoyage.co/>. The rest of navigating through the pages, signup and logins remain the same as you do by running it in the webserver locally. These instructions were already fully discussed in detail in the previous deliverables, the same which can be viewed in the user manual available on the help page <https://medvoyage.co/main/static/docs/UserManual.pdf>. Any further features added after that will be listed and explained in the below section and will be timely updated to the user manual as well.

Deliverable 5 Report

Team Squad 8

E. Clear instructions on how to compile/run both your program and your test case (the program must compile/run)

Prerequisites:

1. Install VS Code
2. Install Docker Desktop

Setting Up:

3. Clone the MedVoyage repository in VS Code.
4. Set up the Python interpreter.
5. Install necessary project dependencies using: ``pip install -r src/requirements.txt``.
 - This installs both ``django`` and ``mysqlclient`` required for the MedVoyage project.
6. We could also set up the same using python virtualenv:
 - Create a virtual environment.
 - Source into that virtual environment.
 - Install the dependencies as mentioned above.

Database Setup:

7. Set up MySQL DB in Docker. Refer to ``tools/setup_mysql_docker.sh`` for guidance.
8. Run the Docker command: ``docker run -p 3306:3306 --name mysqlinstance -e MYSQL_ROOT_PASSWORD='pa$$w0rd' -d mysql:latest``.
9. Access the MySQL instance with: ``docker exec -it mysqlinstance mysql -uroot -p'pa$$w0rd``.
10. When the MySQL command line opens, create the database using: ``CREATE DATABASE MedVoyagedb;``. After creation, exit.

Django Setup:

11. Execute ``python src/manage.py makemigrations`` to create tables and schema based on models in the MySQL database.
12. Apply these migrations using: ``python src/manage.py migrate``.

Running the Program:

13. To launch the Django web server, use the command: ``python src/manage.py runserver``. This starts the server, and we can view the app at ``localhost:8000/``.

Running Test Cases:

14. For executing unit test cases, run: ``python src/manage.py test src/main/tests/``.

Deliverable 5 Report

Team Squad 8

In this deliverable, most of the features that were added were internal bug fixes, email triggers and nothing much of change was made to the webpages or navigation. Thus, the initial setup and user manual remains the same and pretty much like the previous deliverable with few changes in the dropdown menus, email triggers, favicons and buttons which does not change the functionality from the previous deliverable. Anyhow, we tried to include the screenshots for the feature changes that we made for this deliverable for whichever can be demonstrated here.

How to use the basic functionalities of the phase three released features below:

Contact Us: The contact us form is for the doctors to express any questions, comments, or suggestions to the admins by typing in their name, email address and message and clicking on the submit button. The users can view the phone number, contact email address and location.

MedVoyage

About Us Contact Us **Profile** Patient Appointments Sign Out Help

Contact Us

Name:
Enter your name

Email:
Enter your email

Message:
Your message here...

Submit

Phone
(+1) 214-89-22196

Email
medvoyage.service@gmail.com

Location
Discovery Park, N Elm Street, Denton, TX, US.

© MedVoyage

Email trigger response to message: When the user writes a message in the text box provided, immediately a response mail is sent to the account from which the message is sent that the team will get back to them soon.



Deliverable 5 Report

Team Squad 8

Password validation: When the password set does not meet the required conditions, the password is not validated and displays a message for the same.

Password:

.....

- Required Password format: min 8 chars with at least 1 small, capital, number, and special-char each

Booking an Appointment: In this phase, we have developed the book an appointment feature that allows the patient to book an appointment for the doctors available for given dates and the time slots are retrieved from the db according to the availability given by the doctor. The patient can select the date, doctor's name from the dropdown and fill in the details in the text box provided to express their concern.

Book an Appointment

Appointment date:

14-12-2023



Doctor

Dr. Manushree Buyya



Time slot

06:00:00 - 20:00:00



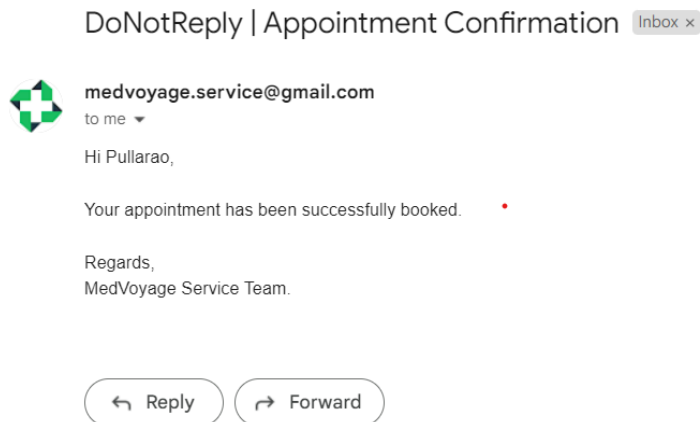
Details

Book Appointment

Deliverable 5 Report

Team Squad 8

Email trigger to appointment booking confirmation: An email is sent to the patient's mail confirming him about the appointment booking. Similarly, email is triggered when sign up is done.



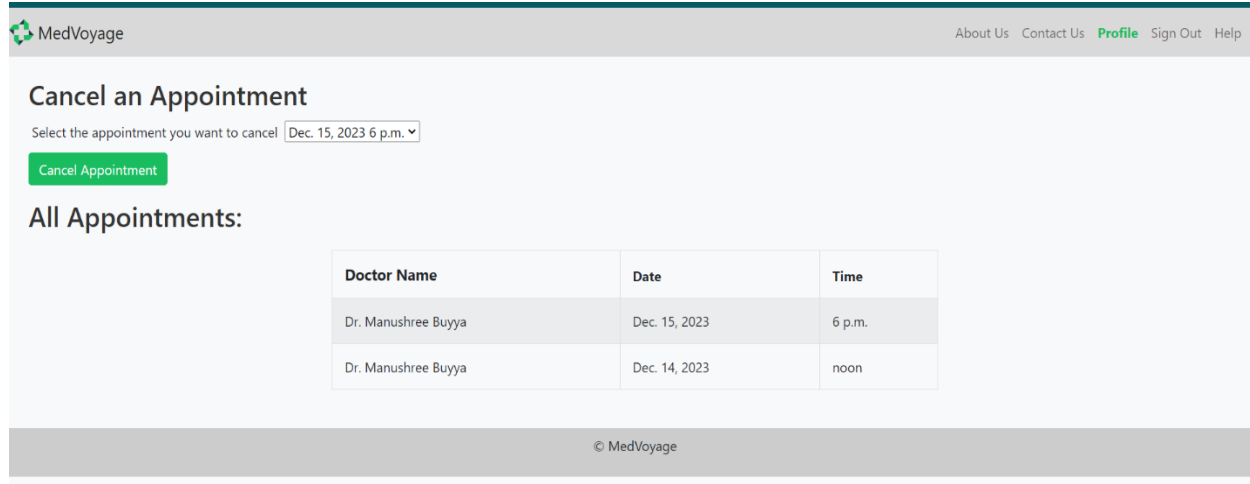
Patients Appointment list: A patient appointments list in the patients panel that shows a list of all the confirmed and booked upcoming appointments of the patient.



Deliverable 5 Report

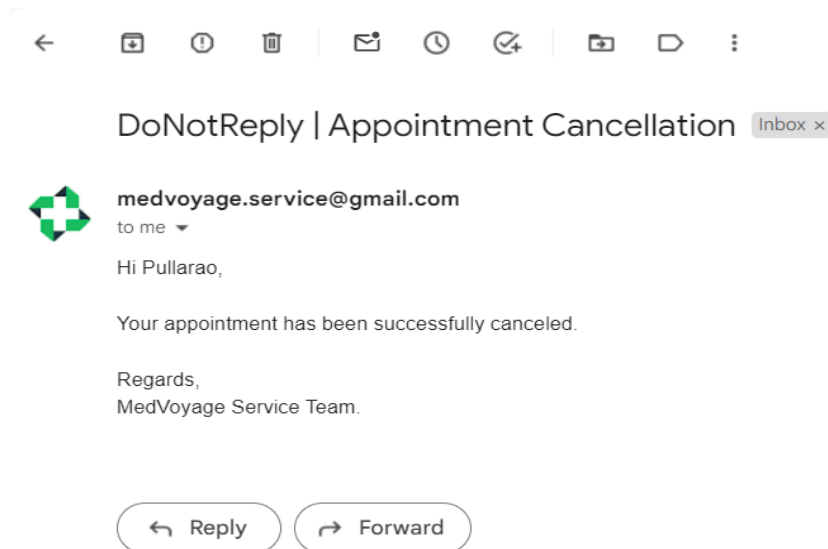
Team Squad 8

Appointment Cancellation: A button was added to the patient's dashboard that redirects the patient to the appointment cancellation page which displays all the appointments booked by the patient. There is a dropdown to choose one at a time and when clicked on the cancel button, the appointment is removed from the list of upcoming appointments and an email is sent as well.



Doctor Name	Date	Time
Dr. Manushree Buyya	Dec. 15, 2023	6 p.m.
Dr. Manushree Buyya	Dec. 14, 2023	noon

Email trigger on Appointment Cancellation: An email trigger is set whereafter successfully cancelling an appointment an email is sent to the user saying their appointment has been cancelled successfully.

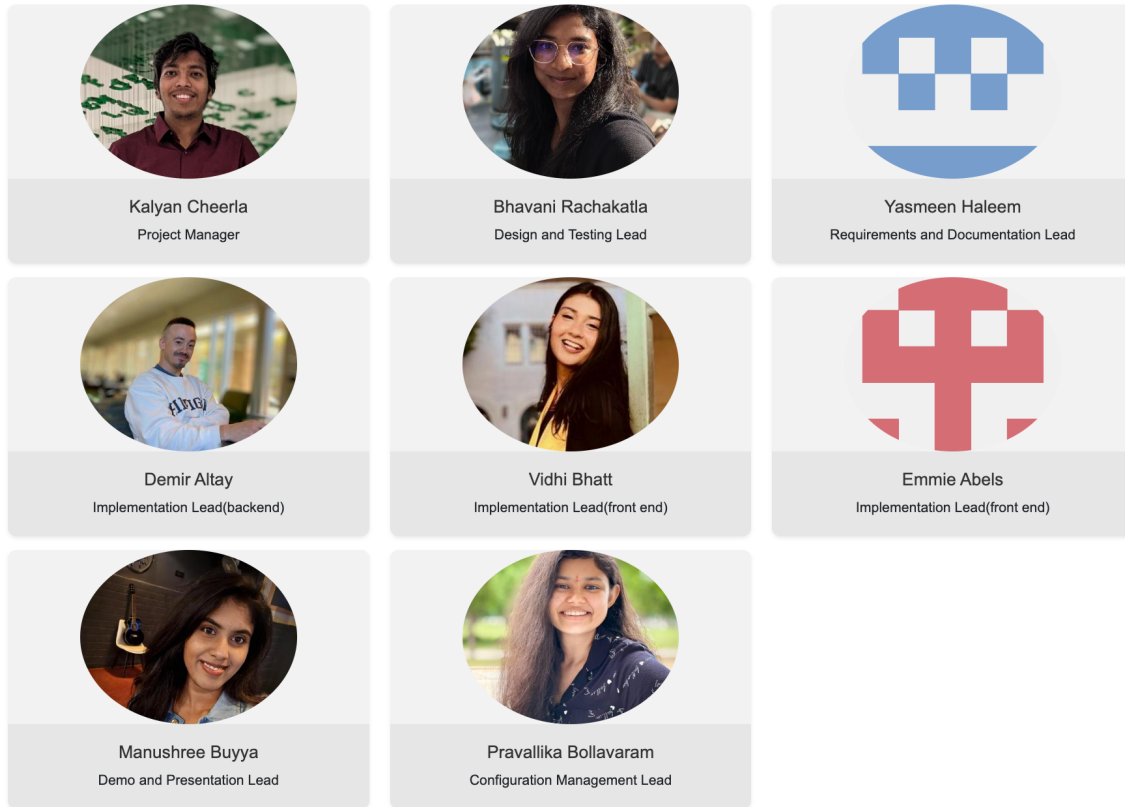


Deliverable 5 Report

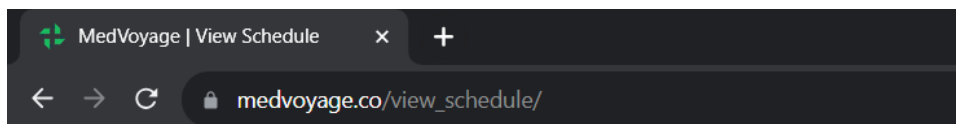
Team Squad 8

About us page: This feature fetches the profile pics from the GitHub account and displays it for the respective developers' profile pic on the about us page.

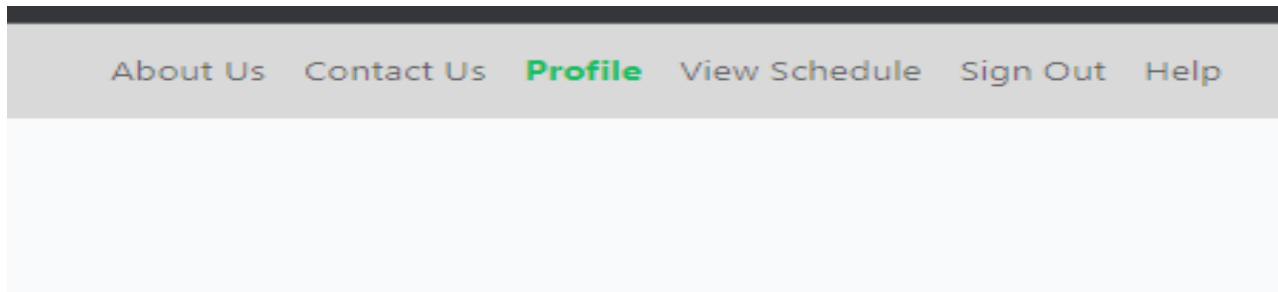
Meet the Developers



Favicon: A favicon file was designed and added to the webapp when opened in the browser. The favicon can be viewed on the browser tabs.

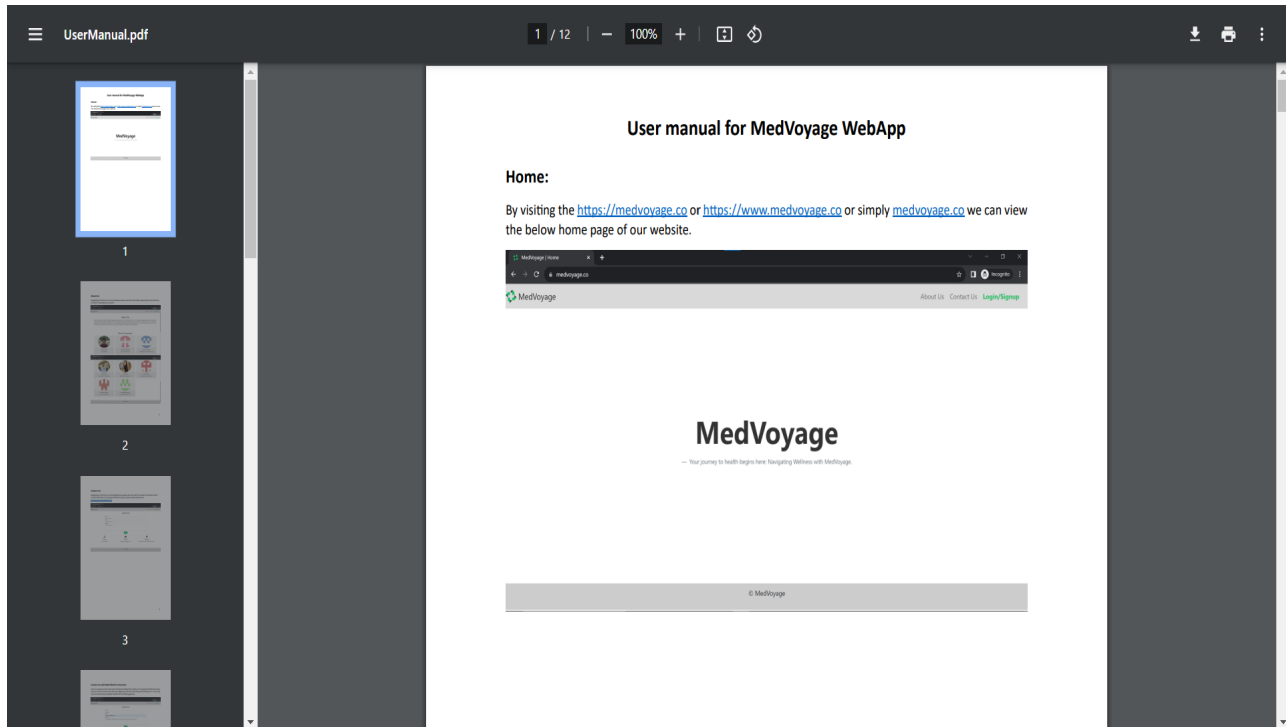


Help on Navigation Bar: A help tab was added to the navigation bar which redirects to a pdf of user manual that opens in the same browser tab that guides the user about the webapp.



Deliverable 5 Report

Team Squad 8



Phone Number Validation: A feature that validates only the correct format of phone number in the phone field during signup. The phone number is indeed an existing phone number that can get text messages for OTP validation.

Email:

Phone:

- Required phonenumber format: '+1555555555' or '555555555' or '555-555-5555'

Security Question:

Deliverable 5 Report

Team Squad 8

F. Ending Feature Summary:

We have implemented almost 90% of the features that we had planned on doing in the beginning that were listed in the first deliverable. Whatever could not be achieved in the given time frame was either modified or pushed to the next deliverable according to the agreement of the team. Here is the list of the features and their progress.

Features that were completed on time:

Feature	Phase
Front Landing page	1
About us page	1
Contact us page	1
Navigation bar	1
Mobile user-friendly design	1
Configure the Database	1
Unified signup form	1
Unified Login form	1
Handle basic different authorizations for patients and doctors	1
Authorization DB integration	1
Forgot Password Feature	1
Sign Out page	1
Password change form	1
Webpage Footer	1
GitHub CI/CD work	1
Doctor Availability Tracker	2
Patient upcoming appointments list	2
Patient previous appointments list	2
Enabling notification capabilities backend	2
Appointment Doctor Confirmation capability	2
Two-Factor Authentication (2FA)	2
DB Table Schema	2
Doctor Appointment Booking System for Patients	3
Contact us backend to trigger emails	3
Fixes for MedVoyage favicon	3
Help link to user manual	3
Book an appointment button quick fix	3
Password validation fixes	3
Frontend logic for booking	3
Frontend fixes for reset password page	3
Frontend fixes for signup	3

Deliverable 5 Report

Team Squad 8

Added unavailability for booked slots	3
Bug fixes on all pages	3
Fetch profile pics on about us	3
Contact us details	3
Colour fixes	3
Fetch doctor details	3
Production bug	3
Adding Docker file	3
Deployment on docker container	3
User manual	3
Backend of Appointment booking system	3
Phone number field validation	3
Login bug Fix	3
Patient appointments list	3
Appointment cancellation	3
Add slots and view slots list	3

Features that were not completed as planned:

Feature	Phase	Comments
Custom Error pages (for HTTP 400, 300, 500 status codes)	1	Not yet started and was pushed to next phase due to time constraint
Client profile update form	1	Was in process by the deadline as we were working on few errors.
Doctor profile update form	1	Was in process by the deadline as we were working on few errors.
Calendar Style widget	2	Was eliminated and not implemented as it doesn't really fit in anywhere and was unimportant.

Deliverable 5 Report

Team Squad 8

Appointment Booking system	2	Started in progress and was completed in the next phase
Doctor day-to-day appointments list	2	Started in progress and was completed in the next phase
Appointment deletion capabilities	2	Started in progress and was completed in the next phase
Basic search to get relevant doctors by name	2	Started in progress and was completed in the next phase
Clear dashboard for doctors	2	Was completed but paused for few changes in the look of the page.
Contact Us backend implementation	2	Pushed to future development as it required more time and effort than estimated
Patient Appointment Deletion Capabilities	2	Pushed to future development
Admin authentication and dashboard system	3	We found an alternative for managing the admin capabilities by using Django admin panel.

Most of the deliverable 5 to do's were not implemented as planned as we found that we can have other alternative for managing the admin dashboard system and authentication using the Django admin panel which is way more simple and easier to implement.

Feature advancements for next phase:

Handling the admin capabilities, data privacy and security concerns can be strengthened in the next phase of the project. We could also integrate the payment part for the application after booking an appointment . Any reschedules and cancellations of the appointment also should be handled which can be an advancement to the current project which we haven't outlined in the manifesto. Prescription handling by the doctor is another such feature which could be implemented as an advancement.

G. Reflection:

Throughout the implementation phase, the team has mostly made good progress on the features planned to be implemented. All the incomplete features that the team was unable to implement on the last deliverable were finished in this deliverable phase. The admin part of the features that were planned to be completed in this phase were not implemented due to change in the approach as we thought of achieving the same with a simpler method, the Django admin panel. The planning stage for these features revolved around utilizing Trello and attempting to divide tasks as evenly and smoothly as possible. MedVoyage's progress was quite significant and valuable from the last deliverable to this one, as the team implemented features that fixed a lot of shortcomings in the previous deliverable and added significant features like email triggers, password validation. As the complexity of some of the features increased from the last phase, some features are still currently in progress. The team recognizes it is important to finish assigned feature implementation. The

Deliverable 5 Report

Team Squad 8

team hopes to increase communication and establish a strong background for Django as currently many of the developers are beginners. In addition to this, the team has consistently dedicated resources to learning and helping any other developers in need through either peer programming or aiding in debugging. MedVoyage hopes to learn various Django frameworks and successfully implement and apply them to the web application. Overall, MedVoyage was a successful and progressive project that has given the team scope for challenges, learning and productivity. Hopefully, we can work on the extension of features of the same project in the near future or the coming semesters.

H. Member contribution table (should describe who wrote what components or classes of the system and what parts of the report).

Member Contribution Table:

Member	Contribution and Overall / Contribution(percentage)
Kalyan Cheerla	Implemented Appointment Booking system, GCP and DNS configuration, deployment using docker and other.
Yasmeen Haleem	Added the phone number and password validation fixes so that it now doesn't accept passwords and numbers on single character that was an issue earlier, contributed with the use case diagrams and class diagrams and sequence diagrams and added the meeting minutes.
Bhavani Rachakatla	Fixed HTML title issue on add slots and slots list. Proper testcases were added for cancellation and appointment booking. Added fix for reversing availability hours on appointment cancellation using unavailable flag. Collaborated with Kalyan on different feature developments, deployments, and discussions. Also reviewed PR#34. Worked on demo video and trimming and presentation file.
Manushree Buyya	Added password validation fixes, worked on email triggering with yasmeen for cancellation of appointments. Worked on the documentation part for the report of phase 3.
Pravallika Bollavaram	Fixed few bugs and issues related to patient appointment booking and dashboard. Handled documentation during deliverable 5 and prepared the Presentation file.
Vidhi Bhatt	Permuted past implementations involving frontend restyle on the patient appointments display functionalities form, effaced patient appointments from the doctor dashboard and the navigation bar, overhauled test cases for implementation, and appended appointments to the patient dashboard
Demir Altay	Fixed a few backend bugs, helped implement cancel appointments. Polished various portions of the frontend.
Emmie Abels	Updated the backend implementation for view schedule and updated the input method on the cancel appointment page. Also helped with front end design and styling for view schedule and cancel appointment.

Deliverable 5 Report

Team Squad 8

Percentage contribution table for Deliverable 5:

Member	Overall / Contribution(percentage)
Kalyan Cheerla	20
Yasmeen Haleem	10
Bhavani Rachakatla	12
Manushree Buyya	12
Pravallika Bollavaram	12
Vidhi Bhatt	10
Demir Altay	12
Emmie Abels	12