
The Art of Computer Programming – A difficult book to understand

Donald Knuth

Dennis Ritchie

B.Sc.(Hons) in Software Development

APRIL 16, 2016

Final Year Project

Advised by: Dr Alan Turing

Department of Computer Science and Applied Physics
Galway-Mayo Institute of Technology (GMIT)



Contents

1	Background	5
2	Introduction and Context	8
2.1	General Problem Statement	8
2.2	Previous work	8
2.3	Purpose	9
2.4	Scope Objective and Goals	9
2.5	Roadmap	10
3	Methodology	11
4	Technology Review	13
4.1	Prototyping Tool - Justin-mind	13
4.2	MEAN-Stack	13
4.2.1	NoSql Technology - MongoDB	14
4.2.2	Express.js	15
4.2.3	Node.js	15
4.2.4	AngularJs	15
4.3	REST- Architecture	16
4.4	Passport JWT - JSON Web Tokens Authentication	16
4.5	Send-grid - Email Delivery	18
4.6	Cross-Platform Development	18
4.7	Google Maps	18
5	System Design	19
5.1	System prototype	19
5.2	Architecture	19
5.2.1	Presentation Layer	20
5.2.2	Business Logic	24
5.2.3	Data Layer	27

<i>CONTENTS</i>	3
6 System Evaluation	31
6.1 Testing	32
7 Conclusion	33

About this project

Abstract A brief description of what the project is, in about two-hundred and fifty words.

Authors Explain here who the authors are.

Chapter 1

Background

[1] Galway City originally formed from a small fishing village located in the area near the Spanish Arch called ‘The Claddagh’ where the River Corrib meets Galway Bay. Galway later became a walled town in the year 1232 after the territory was captured by the Anglo Normans lead by Richard De Burgo. The town walls, some sections of which can be seen today near the Spanish Arch, were constructed circa 1270. A charter was granted in 1396 by Richard II which transferred governing powers to 14 merchant families, known locally as the 14 tribes of Galway. The 14 tribes relished their independence but retained their close links to the British crown. Galway’s strategic coastal location and natural harbour area resulted in a successful trade with both Portugal and Spain and the city prospered for centuries. However in 1651 with the arrival of Cromwell the region entered a long period of decline. Other prominent sea ports emerged on the east coast, namely Dublin and Waterford and trade with Spain came almost at an end. Many years would pass before Galway would again enjoy such prosperity but the legacy of the cities long and colourful history is evident in the character and style of the city. Galway City is a thriving, bohemian, cultural city on the western coast of Ireland. Along with being a popular seaside destination with beautiful beaches and long winding promenade, it also has a buzzing cosmopolitan city centre. The city is a joy to explore with its labyrinthine cobbled streets, colourful shop facades and busy café/ bar culture. The city is also well known for its many festivals throughout the year with huge crowds gathering for the annual Galway Arts Festival, Races and numerous other events. Old Ireland is present too with turf fires and traditional music featuring in many pubs to compliment your enjoyment of a well-earned pint of Guinness. Take an evening stroll along the promenade and watch the sunset over Galway Bay or watch the salmon fishermen in the River Corrib from the perfect vantage point of the Salmon Weir Bridge. [2] Galway is certainly one

of the best tourist attraction in Europe because of its rich cultures, traditions, festivals. Galway is bidding to become the European Capital of Culture in 2020. The bid represents an opportunity for everyone to join their hands together as a community and reflect and spread the uniqueness of our Galway culture and the richness, vitality and diversity of our shared European culture. [3] Failte Ireland has provided the regional tourism performance of 2014. The overseas visitor to counties in 2014 shows that, there were total of 1235 people visiting Galway, and ranks 3rd in Ireland after Dublin (4,119) and Cork (1,542). Failte Ireland has also provided the result of Overseas visitor revenue (€mn) by county in 2014. Adapted from :

Overseas visitors (000s) to counties in 2014

County	Total	Britain	Mainland Europe	North America	Other Areas
Dublin	4,119	1,217	1,730	836	336
Carlow	63	31	21	10	2
Kilkenny	249	46	98	88	16
Tipperary (South)	134	56	41	28	9
Waterford	255	89	69	81	16
Wexford	240	124	67	35	13
Cork	1,542	609	539	293	102
Kerry	1,040	202	371	393	75
Clare	561	140	161	228	32
Limerick	494	193	152	127	22
Tipperary(North)	56	35	12	5	4
Offaly (West)	10	6	3	1	-
Galway	1,235	259	543	335	98

Overseas visitor revenue (€mn) by county in 2014

County	Total	Britain	Mainland Europe	North America	Other Areas
Dublin	1,378	249	572	301	256
Carlow	30	7	12	8	2
Kilkenny	33	9	11	11	3
Tipperary (South)	37	15	13	7	1
Waterford	52	22	12	12	6
Wexford	54	30	15	7	2
Cork	550	134	210	162	44
Kerry	228	54	64	93	16
Clare	128	35	24	61	8
Limerick	169	54	53	52	10
Tipperary (North)	27	13	10	3	1
Offaly (West)	2	1	1	*	-
Galway	350	77	124	120	28

About Galway Civic Trust: [4]Dúchas na Gaillimhe - Galway Civic Trust is committed to protecting and enhancing Galway's natural, built and cultural heritage for the benefit of all. The Trust adopts a hands-on approach and undertakes improvement projects which otherwise would not happen. Our offices are located in the Latin Quarter at the Red Earl's Hall archaeological site. Established in 1992, we have completed over 50 projects, ranging from the erection of historic wall plaques to the refurbishment of Rusheen Bay Bird Sanctuary and restoration of the Fishery Tower at Wolfe Tone Bridge. In partnership with the Department of Social Protection, Galway City Council and Galway County Council, Galway Chamber of Commerce and Industry, The Heritage Council of Ireland, and the local community, we undertake works for the enhancement of Galway city and county.

Chapter 2

Introduction and Context

2.1 General Problem Statement

Galway Civic Trust is an organisation arranging tours around Galway City. They came to us with an issue with which they have been struggling for a while. Any change or addition to their touring program necessitated creating new application or at least updating the existing one. This was impractical as it required Galway Civic Trust to stay in constant touch with GMIT. We have been therefore tasked to develop a cross platform mobile application for the Galway Civic Trust by integrating the backend for content management. The mobile application will be used by the consumer and all the data (tours) will be pulled dynamically from the API. Admin (Galway Civic Trust) can add new tours from the admin panel and all the tours will be synchronised across the consuming devices via Application Programming Interface. The goal of this project is to allow instance synchronisation between the API and the consuming devices so consumer can receive the changes in real time. Developing such an application would certainly solve the issue of Galway Civic Trust. Proposed application will eliminate time consuming visits to GMIT and allow GCT to easily and efficiently make any changes to existing or new tours in the cross platform mobile application through the admin panel.

2.2 Previous work

There were multiple apps developed for Galway civic trust for every new tour. All the content used in the app was static contents therefore It made it impossible to update existing tours or add more tour as needed in the feature.

2.3 Purpose

The whole purpose of this application is to facilitate Galway Civic Trust to CRUD tours or any locations of interests via admin portal. System admin is able to manage the content and this will be reflected in the mobile app. The end user will be able to use their mobile phone to have a look at different places of interests in Galway. The hybrid mobile app will provide end user information regarding the tours. Mobile app can be very useful to tourist travelling to Galway.

2.4 Scope Objective and Goals

The project is separated into three different independent parts. 1: Backed (Admin Panel): Allows system user to perform Create / Read / Update / Delete operation using graphical and interactive user interface. Different roles has been added assign admins, example Super Admin have permission to manage other admin accounts whereas regular admin don't. 2: RESTful API: Allows mobile devices to consume the JSON data. Changes made in the database is reflected by an API and consuming devices automatically receive the changes in real-time. 3: Ionic Mobile App: Cross platform hybrid mobile app has been developed using Ionic framework. App communicates with the API and pull the data.

Objective and Goals: Our objective and goals comes from the previous which work which was carried out by the past Software Development Students of GMIT. Our main objective and goals is to solve the problem stated in the problem statement. Our main goals and objective of this project are as follows:

- Develop Backend for content management.
- Develop Restful API so that the client app can consume data in real time.
- Cross platform hybrid mobile app for consuming the API.
- Google Maps navigation and direction services.
- Instant synchronization of data between API consuming devices and backend.

2.5 Roadmap

So, at this point, we have discussed about what the project is, and the reason for taking this project. Now, let me explain few sections/chapters of this review briefly so you will have a better understanding of a project. Methodology - This section provides the development methodologies we used to develop this project, including project meetings, collaboration tools used, interaction with the client and weekly meeting with the project supervisor. Technology review : This section provides the research that was carried out when choosing the technology . This section also provides the selection criteria and the particular reason for choosing the specific technology. System Design : This section gives detailed information about actual system, and the way it is functioning. This section also talks about the architecture of a project, Data models, use cases etc. System Evaluation : This section provides evaluates the system based on the following attributes. Scalability Robustness Maintainability Extensibility Reliability

Conclusion and Recommendation: This section summarises the context of a project, objectives and goal achieved. We have talked about different sections of this project briefly and now I would like to include the resources url of this project. Due to the nature of this project, we have created three different repositories in Github as because they were developed independently of one another and at the end they are talking using the middleware technologies and different protocols. This is explained in detail in the technology review section. Following are the repositories in the github.

Chapter 3

Methodology

About one to two pages. Describe the way you went about your project:

- Agile / incremental and iterative approach to development. Planning, meetings. (2 para)
- What about validation and testing? Junit or some other framework. 2(para)
- If team based, did you use GitHub during the development process. (1 para)
- Selection criteria for MEAN stack and Ionic Framework. (10 lines)

Check out the nice graphs in Figure 3.2, and the nice diagram in Figure ??.

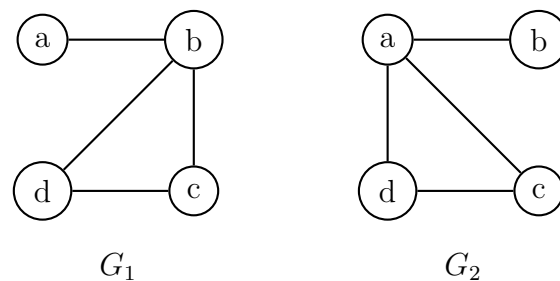


Figure 3.1: Nice pictures

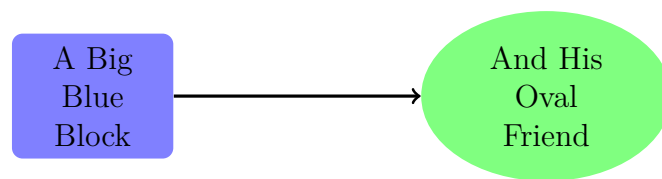


Figure 3.2: Nice pictures

Chapter 4

Technology Review

In this section we are going to talk about the technology used in this project. We have used various types of new technologies. This sections will cover about the technology choices and the reason for choosing the technologies. We have used MEAN stack for backend development, Ionic Framework for cross platform app development and JustInMind as a prototyping tool. We are first going to talk about prototyping tool then MEAN stack and finally we will look at the ionic framework and cross platform development.

4.1 Prototyping Tool - Justin-mind

JustInMind in All-In-One prototyping tool for web and mobile app. [5] It allows you to provide prototype faster and better with default built-in widgets. These widgets have been specifically designed to fit our IOS or Android app, so you content always looks great. JustInMind also provide the web framework to prototype web applications. Every widget in web framework is designed to help your web wireframes. JustInMind includes handy collection of web buttons, menus, charts and more that can be used automatically. JustInMind also gives us an ability to embed HTML and Documents. The widget library allows you to insert HTML, Docs and video, including online widgets, interactive maps and other on/offline content, to name a few.

4.2 MEAN-Stack

So, what exactly is MEAN stack ?Well, in nutshell [6] MEAN is a free and open-source JavaScript software stack for building dynamic web sites and

web applications. The MEAN stack makes use of MongoDB, Express.js , Angular.js and Node.js. Because all components of MEAN stack support programs written in JavaScript, MEAN applications can be written in one language for both server-side and client-side execution environments. In order to fully understand the use and why MEAN stack is so popular and one of the mostly used javascript software stack, we must talk about its components. Let's start with M (MongoDB) of MEAN stack.

4.2.1 NoSql Technology - MongoDB

MEAN stack comes with a NoSql database technology called MongoDB. Before talking about MongoDB itself it is important that we understand what NoSql technology is about. [7] Not only Sequential Query Language(NoSql) is a database that provides a mechanism for storage and retrieval of data which is modeled in means other than the tabular relations used in relational databases. NoSql databases have been proven to be the solution to what is known as Big Data as they follow a schema-less data model, hence provide increased scalability and flexibility as compared to relational databases. In recent years, developers and organization have experiences a sharp rise in volume of user data and products that has to be stored in databases [8]. NoSQL databases are widely used to store and retrieve very large amounts of data using a key-value format [9]. These types of databases have emerged as the best choices that suite modern mobile and web development. We discussed briefly about the NoSql database technology. Now let's briefly talk about MongoDB .

MongoDB : This is a document store, non-relational, open-source database developed by 10gen. The name mongo is extracted from the word humongous. It provides high availability , high performance, and automatic scaling and allows data insertion without a predefined schema. A record in MongoDB is composed of field and value pairs and are predefined schema. A record in MongoDB is composed of field and value pairs and are similar to JSON objects. The value of field may consists of arrays, and array of documents or other documents. MongoDB maintains data consistency in the sense that one write operation to the data in the database allow subsequent read operation. They use a locking mechanism that contributes to increased execution time as the number of update operation increases. [8] [10].

Anam Zahid , Rahat Masood , Muhammad Awais Shibli, in their paper describe how MongoDB offers horizontal scale-out for databases using a technique called sharding. With sharding, a data is distributed across multiple physical partitions known as shards. This was designed in order to address the hardware limitations where only a single server existed and

contributed to such things as bottlenecks in RAM or disk I/O. MongoDB has the sharding functionality automatically built into the databases and the size of the data grows, MongoDB automatically balances the data in the shard and so when the size of cluster decreases or increases. As a result , a dynamically balanced load is experienced. Concurrency control measure for multiple clients accessing the same database are enforced by mongoDB by managing multi-threaded access to shared objects and data structures. [8] [11].

4.2.2 Express.js

[12]Express.js is another component of MEAN Stack. Express is a nodejs based web framework, inspired by Sinatra and it is asynchronous. Express.js builds on the underlying capability of Node, by providing a web application server framework. Express.js is a Node.js web application server framework, designed for building single-page, multi page and cross platform hybrid web applications[13].

4.2.3 Node.js

Node.js is a Javascript runtime built on Chrome's V8 JavaScript Engine. Node.js uses an event driven, non-blocking I/O model that makes it lightweight and efficient web server environment, idea for constructing a web-service API's[14]. Node.js package ecosystem, npm, is the largest ecosystem of open source libraries in the world. [13]

4.2.4 AngularJs

The last component of a stack is Angularjs. The official documentation [15]defines angularjs as - AngularJS is a structural framework for dynamic web apps. It lets you use HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and succinctly. Angular's data binding and dependency injection eliminate much of the code you currently have to write. And it all happens within the browser, making it an ideal partner with any server technology.

Angularjs provides client-side framework for MVC [15] single page web application. To take full benefit of Angular, it can be well be used with other software packages like Yeoman and Bootstrap. Yomean provides an environment which enables the use of generators: simple script-based tools that can be used to scaffold the bare-boles of Angular web app. [16]. There are several reason why angularjs is so popular and few reasons includes:

1: As mentioned above, Angularjs structure the source code by following the Model View Controller. 2: Another power feature of Angularjs is the capability of two way data binding. It lessens the amount of boilerplate code which is written to keep the model and view in agreement. 3: Angularjs models are plain old java object (POJO), therefore it is simple to change or append properties without any complications. 4 : Another most important feature of Angularjs js is about dependency injection. Dependency injection is a software design pattern that deals with how components get hold of their dependencies. The angular injector subsystem is in the charge of creating components resolving their dependencies, and providing them to other components as requested [17]. So, we spoke about the main components of MEAN stack briefly and now you should have better understanding of this technology. Lets us now talk about the RESTful api that as been developed for the consumption of our data by the mobile device(end user).

4.3 REST- Architecture

[13][18] Representation State Transfer, widely knows as the REST or RESTful model for web-services, uses the native HTTP operation: POST,GET,PUT, and DELETE to map on to the four fundamental database operations. - Create,READ,UPDATE,DELETE. API can be built to link these four HTTP verbs to functions which Create, Read, Delete or Update records within a web-services. This service can be consumed by any time of authenticated client device.

4.4 Passport JWT - JSON Web Tokens Authentication

The backend(admin portal) and the API is fully protected using JSON web tokens. System user Login and API endpoints has been all protected from external injection. [19] [20] [21] [19,20,21] JSON web tokens are open, industry standard RFC 7519 method for representing claims securely between two parties. Token based authentication is prominent everywhere on the web nowadays. With most every web company using an API, tokens are the best way to handle authentication for multiple user. JWT is separated into three parts, Header, Payload, Signature.

Header: The header contains two parts - declaring the type, which is jwt and the hashing algorithm to use (HMAC SHA256) “Typ”: ”JWT” “Alg”: ”HS256” 2: Claims : The payload will carry the bulk of our JWT,

also called the JWT claims[19]. This is where we put the information that we want to transmit and other information about our token. The claims contains any information that you want signed[22]. 3: JSON web Signature: The headers are claims digitally signed using the algorithms in the specified in the header. The header and claims are JSON that are base64 encoded for transport. The header, claims, and signature are appended together with a period character [20] [21]. **The JWT authentication is handled using passport module that is available from npm. This module lets you authenticate endpoints using a JSON web token. It is intended to be used to secure RESTful endpoints without sessions.**

[23] One great advantage of tokens is that we don't have to lookup the token in a database on every api call as it contains all needed information in itself. This should help keeping the authentication process small. The biggest downside of that is the inability of revoking a token without having a whitelist or blacklist somewhere. This is the reason we keep the lifetime of the token small (120 minutes in this example)

Here is an example configuration which reads the JWT from the http Authorization header with the scheme 'JWT' [24]:

```
var JwtStrategy = require('passport-jwt').Strategy,
    ExtractJwt = require('passport-jwt').ExtractJwt;
var opts = {}
opts.jwtFromRequest = ExtractJwt.fromAuthHeader();
opts.secretOrKey = 'secret';
opts.issuer = "accounts.examplesoft.com";
opts.audience = "yoursite.net";
passport.use(new JwtStrategy(opts, function (jwt_payload, done) {
    User.findOne({id: jwt_payload.sub}, function (err, user) {

        if (err) {
            return done(err, false);
        }
        if (user) {
            done(null, user);
        } else {
            done(null, false);
            // or you could create a new account
        }
    });
}));
```

4.5 Send-grid - Email Delivery

The process of sending email to system admin with instructions to reset the password has been achieved using SendGrid API. SendGrid provides a cloud-based email delivery service that assists business with transactional email management [25]. SendGrid is chosen in this project for password recovery email delivery service because it is very reliable and have advanced spam filtering system. We got sendgrid for absolute no cost for development purpose.

4.6 Cross-Platform Development

End user mobile app has been developed in Ionic Framework. The requirement of this project was to target at least two major platform, Android and IOS. Ionic framework was a perfect choice for us because It is free and open source, Ionic offers a library of mobile-optimised HTML, CSS, and JS CSS components, gestures, tools for building highly interactive apps. Built with Sass and optimised for AngularJs [12].

4.7 Google Maps

For GPS navigation and directions of a tour in consumer (ionic app), google Direction service is used. It allows you to calculate directions by using the DirectionsService Object. Adapted from [26], the official documentation .This object communicates with the Google Maps API Directions Service which receives direction requests and return computed results. You may either handle these directions results yourself or use the DirectionsRenderer object to render these results.

About seven to ten pages.

- Describe each of the technologies you used at a conceptual level. Standards, Database Model (e.g. MongoDB, CouchDB), XML, WSDL, JSON, JAXP.
- Use references (IEEE format, e.g. [1]), Books, Papers, URLs (timestamp) – sources should be authoritative.

Chapter 5

System Design

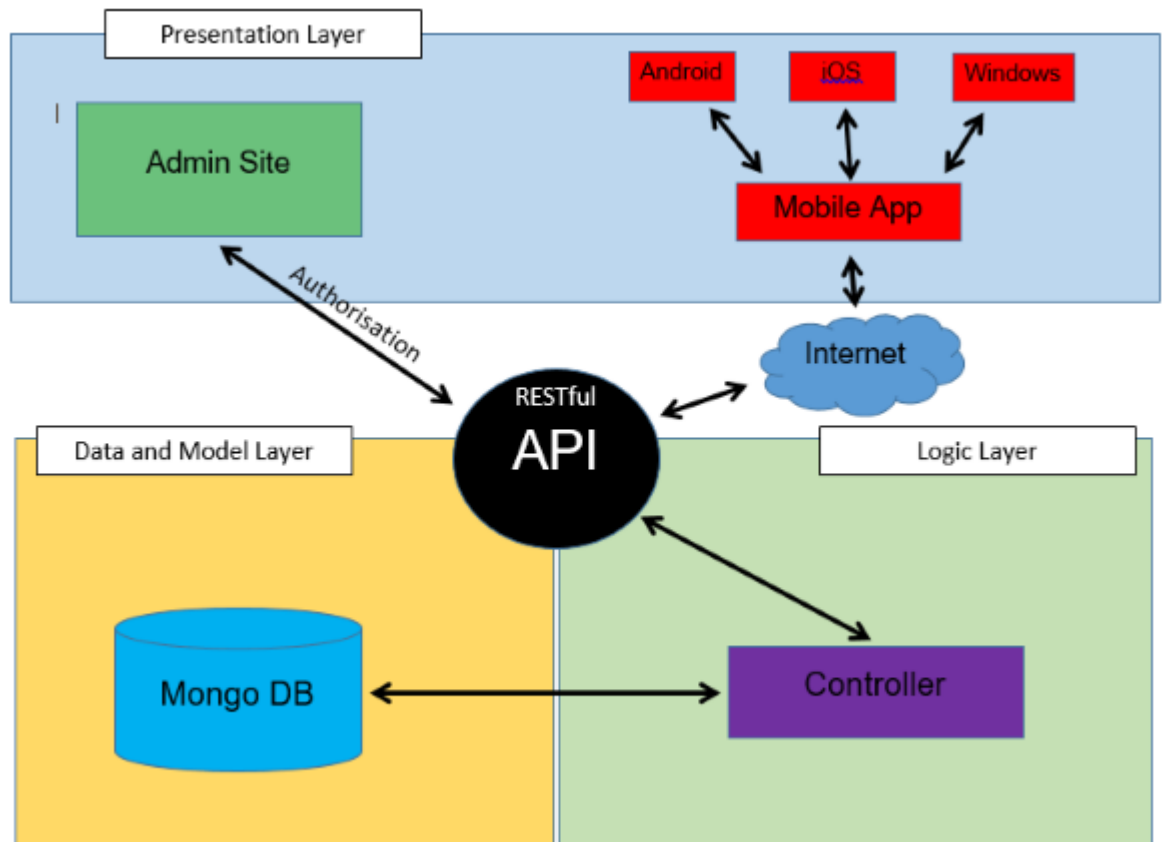
As many pages as needed.

5.1 System prototype

Prototyping tools used : Reasonable amount of time was spent initially in the design phase of this application . We have designed our both backend (Admin portal) and Ionic mobile app using , one of the most popular prototyping software called JustInMind Prototyper.

5.2 Architecture

Architecture : The application is separated into three layers. Presentation layer, Business logic layer and Data access layer.



5.2.1 Presentation Layer

In presentation layer you can see the Admin Site (Admin portal) ,this is a graphical user interface where system admin can manage the contents and accounts of the users registered. On the right hand side, you can see many mobile devices of different platform consuming the data from API. Consuming app is only allowed to send GET request to API because all we care in end user app is being able to retrieve the data dynamically loaded from the api. On the admin portal, any CRUD operations needed to perform requires authorization and authentication of registered admin. The user interface in presentation layer was developed in angularjs and Ionic framework. The main components of presentation layer are as follows : In this layer will will discuss about the admin portal user interface and end user mobile apps user interface.

Login

Email

Password

Login

Forgot Password

Login screen for system admin. If admin forgot his password, it can easily be reset using forgot password link . The system will check and verify that the specified admin exist in the database and it exists, system will generate a random temporary password and send an email.

HomeUser ManagementSystem User Registration

Welcome, Arjun Khare!

Total Tours : 3


Add New Tour



Awesome tour.

Tour Name

DeleteEditCurrently Live



Awesome

Test 3

DeleteEditNot yet published



This tour is cool

WaterWays

DeleteEditCurrently Live

This is the landing page of admin panel. When admin logs into the website successfully they will then be brought to this main page where they can see all the tours they have published. The tour also represent its state, ie. published or Not yet published. Admin can easily from CRUD operations.

LOCATION page

Add New Location


View/Add Existing Locations

#	Location Name
1	Museum
2	Wolf Tone Bridge
3	Fisheries Tower
4	Salmon Weir Bridge
5	Poor Clares Convent
6	Parkavera
7	Galway Cathedral
8	King's Gap
9	Nora Bernacle House
10	O'Brien's Bridge

Location Title
Museum

Location
1

Description
Starting outside the modern Galway City Museum the first thing we notice before embarking on our walk is the remains of the city's medieval walls. The walls surrounded the city, protecting it from attack as the fourteen families or 'Tribes' of Galway held political and commercial sway within their confines.

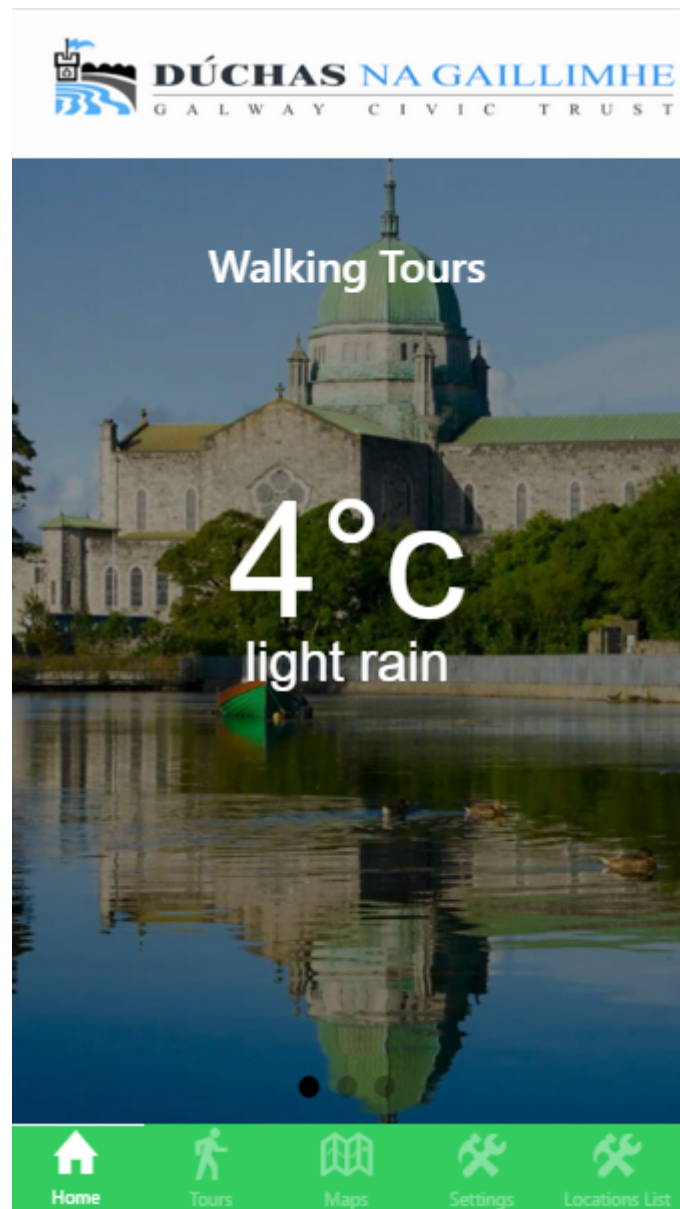


When you click on the tour in the main page, you will be brought to this (location page) where you can see all the locations/walks within that tour. System admin can perform CRUD operations. USER MANAGEMENT PAGE

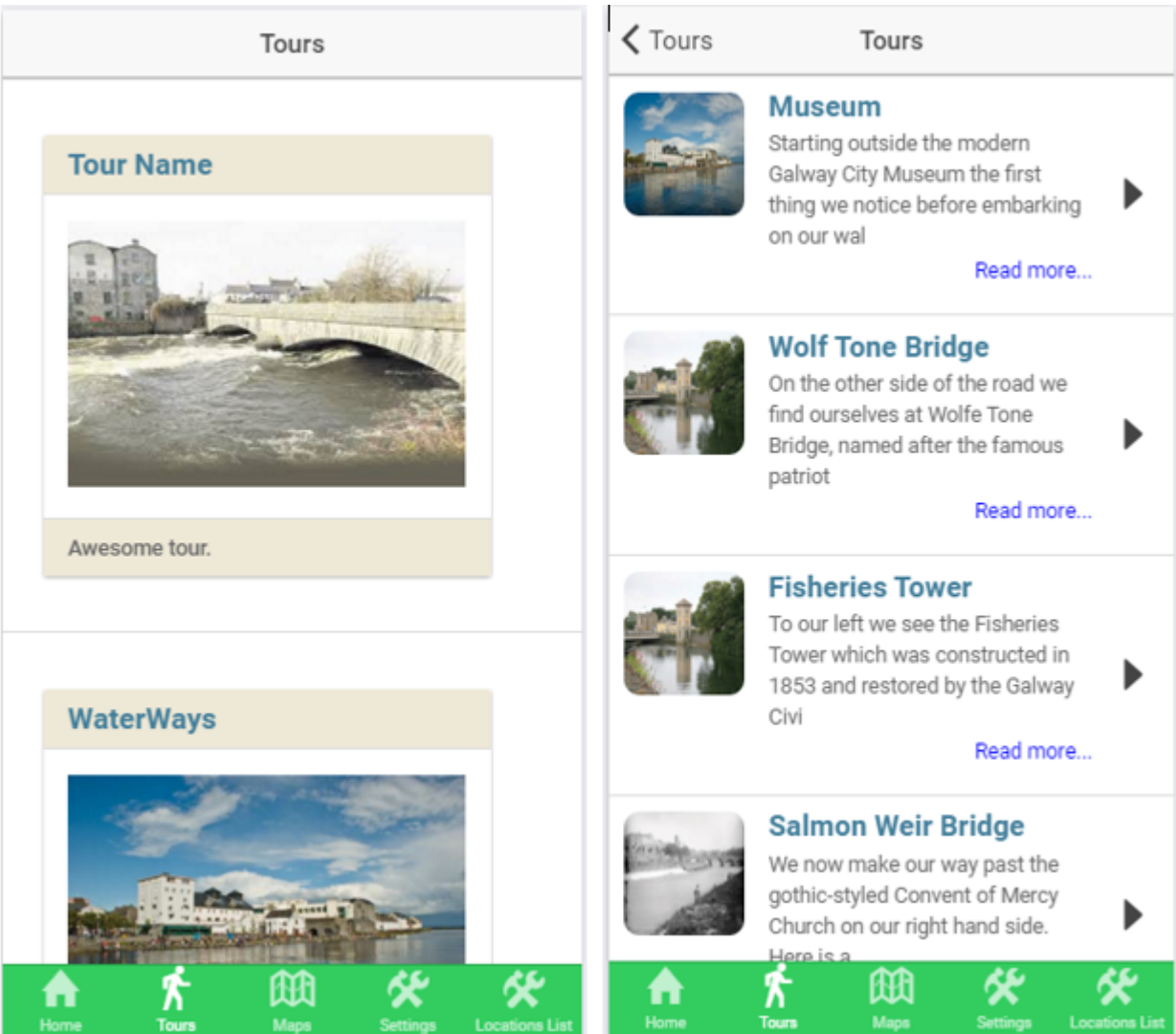
Home
User Management
System User Registration

User Name	User Email
Arjun Kharel	admin@gct.ie
Arjun Kharel	adgn353@gmail.com
Trevor Davies	trevor.daviesgti@gmail.com

This is one of the powerful feature of this project. This is a user management section. Remember, there are two types of admin roles in this project. Super admin and Regular admin. If you are logged in as a superuser then you will have access to user management, system user registration pages.



This is the landing page of the mobile application. The application is very clean and attractive. It shows the weather and on the bottom it have different tabs for navigation purpose.



When user clicks on the tours, they will be brought to the page on the left. End user can see all the tours available. When they click on a tour, they will be brought to the location page where it displays all the location or walks in that tour.

5.2.2 Business Logic

We have talked about the presentation layer of this application . Now let's talk about the business logic layer and how everything is connected. The business logic has been written in javascript. Every functionality has its own controller. When consumer app send the GET request to an API endpoint,

appropriate controller is called to complete the request. The thing to remember here is, external user / external request is not directly connected to our database. The controller is used to retrieve the information from the database and response is sent back to the browser. There is no direct connection to the database therefore any request coming is handled by an API and controllers working together. Please refer to the architecture diagram above for better understanding of this process. API route and appropriate controller is invoked to complete the request.

How tour is uploaded in the database ? 1: Angularjs Controller 2: Send post request to backend API 3 : API now insert data in mongodb..

1: AngularJs upload tour controller :

```
$scope.uploadTour = function(file) {
  console.log($scope.tour);
  Upload.upload({
    url: '/api/tour/upload',
    headers: {
      'Content-Type': 'multipart/form-data'
    },
    data: {
      file: file,
      title: $scope.tour.title,
      description: $scope.tour.description,
      status:$scope.tour.status,
      _creator: $scope.user._id
    }
  }).then(function(resp) {
    alertSuccess.show();
    console.log('success ' + resp.config.data.file.name + 'uploaded. Respon
    $scope.tours.splice(0, 0, resp.data);
    $scope.tour.title = '';
    $scope.picFile = '';
    $scope.picPreview = false;
    alertSuccess.show();

  }, function(resp) {

    alertFail.show();
  }, function(evt) {
    var progressPercentage = parseInt(100.0 * evt.loaded / evt.total);
    console.log('progress: ' + progressPercentage + '% ' + evt.config.data.
```

```

    });

    }//uploadtours
    toursAPI.getAllTours();
  }

exports.upload = function(req, res) {
  var newTour = new Tour();
  var fileimage = req.middlewareStorage.fileimage;

  console.log("The description is :"+ req.body.description);
  newTour.image = '/assets/images/uploads/' + fileimage;
  newTour.title = req.body.title;
  newTour.description = req.body.description;
  newTour.status = req.body.status;
  newTour.createTime = Date.now();

  newTour.save(function(err, tour) {
    if(err) {
      console.log('error saving tour');
      return res.send(500);
    } else {
      console.log(tour);
      res.status(200)
        .send(tour);
    }
  });
};

```

POST and PUT routes protected

```

router.post('/upload', auth.isAuthenticated(), controller.upload);
router.post('/updateTour', auth.isAuthenticated(), controller.updateTour);
router.post('/updateCurrentLocation', auth.isAuthenticated(),
router.post('/uploadLocation', auth.isAuthenticated())

```

Controller to receive all the locations of a specific tour and sorting by timestamp to determine the order of the tour inserted

```

exports.getAllPublishedTours = function(req, res) {
  Tour.find({status:true})
    .sort({

```

```
        createTime: -1
      })
      .exec(function(err, tours) {
        if (err) {
          return handleError(res, err);
        }
        if (!tours) {
          return res.send(404);
        }
        console.log(tours);
        return res.status(200)
          .json(tours);
      });
    };
  }
```

5.2.3 Data Layer

Mongoose Models - MongoDB

Location	Tour	Admin
Location ID : String	Tour ID :	Admin ID : String
Title : String	Title: String	Name : String
Location Address : String	Description : String	Email : String
Image : String	Image : String	Password : String (Encrypted)
Tour ID : Array	Status : Boolean	Role : String
Shared Counter : Number		
Latitude : Double		
Longitude :Double		
Creator : ObjectID		
Create Time : Date		

The back-end comprises of 3 collections. Locations , Tour and Admin collection. The model is represented as a mongoose model. Mongoose provides a straight-forward, schema-based solution to model your application data. It includes built-in type casting, validation, query building, business logic hooks and more, out of the box. Lets talk about the each of field in the collections (models) and about their purpose.

Location: - Lets us talk about the important field in the location collection. Location ID : is a unique ID assigned by MongoDB, TourID is an array that contains the ID of the tour. This is very important in this project because it is used to identify the location-tour relationship or in another words, to find out which tour the particular location belongs to. Data structure is Array because a location can be shared in one or many tours and we don't want to replicate the same data more than once. By having Tour ID array, we can keep track of the location and in how many tours it is being shared. The Shared Counter is another field that determine, how many tours is currently using that location. This is useful when we want to share the location is multiple tour.

Location model representation in Mongoose:

```
var mongoose = require('mongoose');
var Schema = mongoose.Schema;

var LocationSchema = new Schema({
  image: String,
  title: String,
  tourId : String,
  tourIdArr : [String],
  sharedCounter: Number,
  description: String,
  location: String,
  xCoordinate: String,
  yCoordinate: String,
  _creator: {
    type: Schema.ObjectId,
    ref: 'User'
  },
  createTime: {
    type: Date,
    'default': Date.now
  }
});
```

```
module.exports = mongoose.model('LocationSchema', LocationSchema);
```

Tour: - Tour collections contains basic information about a tour. Title represent a tour title, status identifies if the tour is currently published or still under the development. Only with the status "published" tours are synchronised to the consuming devices.

Tour model in mongoose

```
var mongoose = require('mongoose');
var Schema = mongoose.Schema;

var TourSchema = new Schema({
  image: String,
  title: String,
  description: String,
  status: Boolean, 'default': false,
  createTime: {
    type: Date,
    'default': Date.now
  }
});

module.exports = mongoose.model('Tour', TourSchema);
```

Admin: - This collection is the admin collection. This contains the details of system admins. The model of this collection in mongoose looks like this.

```
var mongoose = require('mongoose');
var Schema = mongoose.Schema;
var crypto = require('crypto');
var UserSchema = new Schema({
  name: String,
  email: {
    type: String,
    lowercase: true
  },
  role: {
    type: String,
    default: 'user'
  },
});
```

```
        hashedPassword: String, //hashed password  
        provider: String, //authentication provider  
        salt: String  
    });
```

Chapter 6

System Evaluation

The problem we were trying to solve that is mentioned in the problem statement of introduction section is solved. Back-end is fully functional and tested using WebDriver java. We have successfully implemented what we initially wanted to do. Back-end (Admin Portal) is the hardest part of this application development, and the reason is, all the content management, authentications, and API creations is done in the back-end side of this system. The end user app built in ionic framework is simply consuming the data from an API, and the synchronization is very quick and is working effectively.

In this section we are going to evaluate our system based on different attributes. They are as follows :

- Scalability
- Robustness
- Maintainability
- Extensibility

Scalability: : The system is highly scalable when the requirement grows or the data in the database increases in the near future. MongoDB is used to store the datasets and it is widely used in the Industry. NoSql database technology and scalability issues is resolved easily.

Robustness: The system is very robust and works without any issues. The system has been tested in Selenium Webdriver. Various test has been conducted to check the strength of the system. Test cases are available in the github project. The routes/endpoints in the backend are fully protected from any external injection, using the JWT - passport.

Maintainability - This documentation provide clear understanding of the project. Future maintenance should not be a problem because it is very

well structured. Model view Controller patterns was used throughout the development and everything is well structured in cohesive manner. The code is highly documented and it doesn't impose future maintenance issues.

Extensibility - The development of API makes it possible for many applications to consume the data. If Galway Civic Trust want to use the tour information in different project/system, they do not have to create or insert the same data in another application. The data can be easily shared using the API we built in this project for the consumption by mobile devices. Lets say, if GCT want to add one more feature in the future, developer can easily insert a route code a controller to do that specific task. There is absolutely no need and will never have to mock around with other functionality to implement a new feature. No technical debt has to be paid back to add new functionality because the project was developed by adapting various software design principles and design patterns.

6.1 Testing

Talk about selenium web-driver Java.

- Prove that your software is robust. How? Testing etc.
- Use performance benchmarks (space and time) if algorithmic.
- Measure the outcomes / outputs of your system / software against the objectives from the Introduction.
- Highlight any limitations or opportunities in your approach or technologies used.

Chapter 7

Conclusion

About three pages.

- Briefly summarise your context and ob-jectives (a few lines).
- Highlight your findings from the evalua-tion section / chapter and any opportuni-ties identified.

Bibliography

- [1] “Galway tourism.”
- [2] “Galway2020.”
- [3] “faiteireland.”
- [4] “Galway civic trust.”
- [5] “Just in mind prototyper.”
- [6] “Mean - stack.”
- [7] “Nosql technology.”
- [8] “Mongodb - manual.”
- [9] “Open journal of databases (ojdb) volume 1, issue 2, 2014 which nosql database.”
- [10] “Eric redmond jim r. wilson (2012); seven databases in seven weeks.”
- [11] “Mongodb architecture guide..”
- [12] “Andrew john poulter, steven j.jonston, simon j.coxt.”
- [13] “Express.js wiki.”
- [14] “Nodejs wiki.”
- [15] “Angularjs - official website - introduction section.”
- [16] “Yeoman generator.”
- [17] “Angularjs dependency injection - official website.”
- [18] “Restful programming.”

- [19] “The anatomy of a json web token.”
- [20] “Internet engineering task force rfc 7519.”
- [21] “Draft ieta oauth json web token.”
- [22] “Jwt - the right way.”
- [23] “Token based authentication - nodejs.”
- [24] “Passport - jwt strategy.”
- [25] “Send grid - email service.”
- [26] “Direction service - google maps.”