COMP4920

Project Plan

By Mitchell Biggart, Matthew Geary, Qiwei Yang, Raycole Dai, Prasad Salaskar

Table of Contents

Introduction

This report will describe how the team plans to build Sesalta - an educational game focused on geography.

In particular, this document contains a description of the expected final product, technology and tools to be used in its development, and details regarding team organisation.

Motivation

A well-defined geographical understanding allows individuals to be more in touch with their place in a global society, and early geography education has demonstrable positive effects on students' understanding and empathy for diverse places and cultures (Harrison 2017). Unfortunately, several studies have shown that geographical knowledge areas are - on average - severely lacking, and geography has taken a less prominent role in schooling.

As such, our aim with this project is to develop a game that acts as an easy-to-use platform for learning these key knowledge areas in a fun and engaging way.

Product Outline

This team will produce a geography-based game that can be played in a user's browser.

Initial Product

The initial release will be a single-player game consisting of two modes:

1. One mode which challenges the user to identify the name of a country highlighted on a world map
2. One mode which challenges the user to select a country on a map given its name

After selecting one of these two modes, the user will be presented with a number (currently 5) of questions of that type. They will then receive a score based on the number of questions that they answered correctly.

After each question, the user will see whether they selected the correct answer. If they did not make the correct choice, the correct answer will be shown.

Additional Features

Additional features will be incorporated into further releases of the game. These features are given, very roughly ordered from higher priority (first) to lower priority (last):

- Further releases will include additional game modes, based on national flags and the names of capital cities.
- A second language (Japanese) will be added alongside English.
- Scoring will be modified to take into account the time taken for the user to answer. A time limit will also be added (currently 12 seconds) after which no answer will be accepted. This is intended to limit cheating.
- A personal scoreboard will track a user's game scores for each game mode. This will be available to the user.

- A global leaderboard will track the game scores of players who opt in. Top scores will be visible to all users. Players will be required to set a 3-character name which will be displayed alongside their scores.
    - A filtering system will prevent users from setting names that may be deemed rude or offensive.
    - Users viewing the leaderboard can choose to view the top scores from all-time, the previous week and the previous 24 hours.
- A difficulty setting will allow players use existing gameplay data to determine which countries are more or less widely known. This data will be used then allow players to select a difficulty.
- A trophy system will allow users to collect in-game objects recognising certain achievements, such as obtaining a top-10 score on the weekly leaderboard.
- A login system will provide the option for players to store their scores and achievements, accessing them between devices.

Target Market

Age

The team believes that any person with the desire to enhance their geography knowledge should be given that opportunity with Sesalta. Hence, the game's content is expected to be appropriate for people of almost all ages.

While some games are inappropriate for children (e.g. due to violence or mature themes), games targeted specifically towards young children may deter adults who consider playing.

Therefore, the team intends to design the game such that it is appropriate for young people - around the age of ten. Younger children may then also be able to play the game with the guidance of someone older. The interface should also be void of elements that may deter adults.

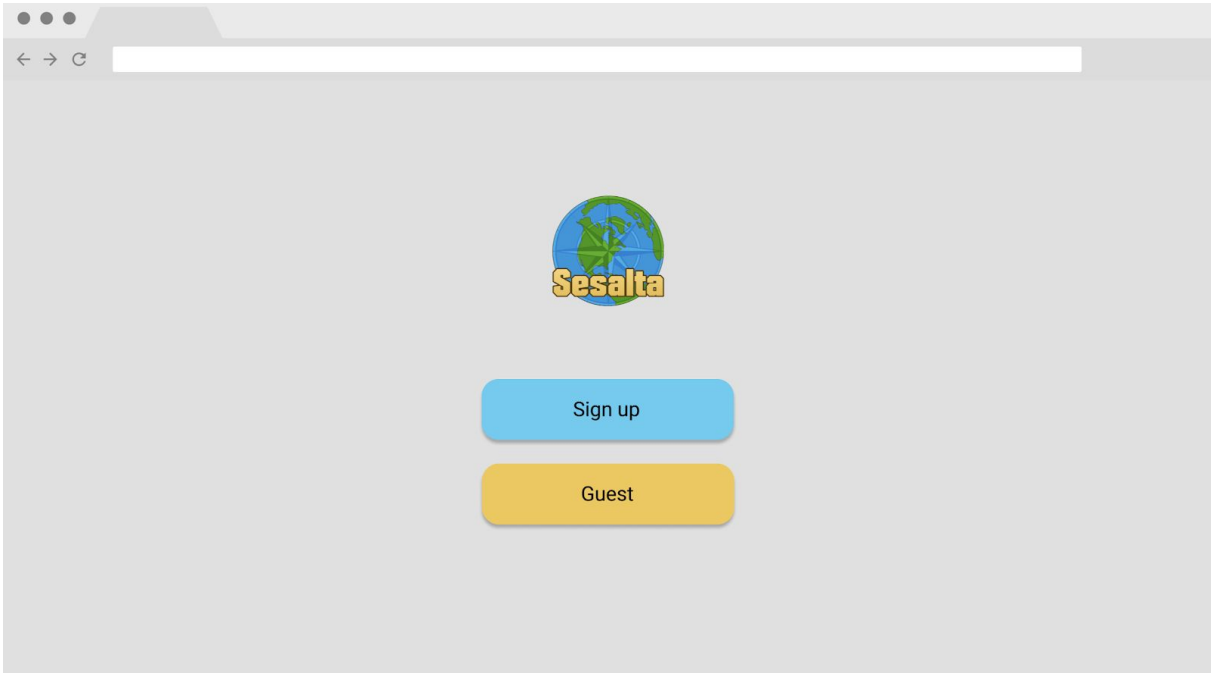
Location and Language

Sesalta will be made available initially in English (MVP). However, the team recognises that language restrictions may limit the game's market and diminish its ability to reach a global audience.

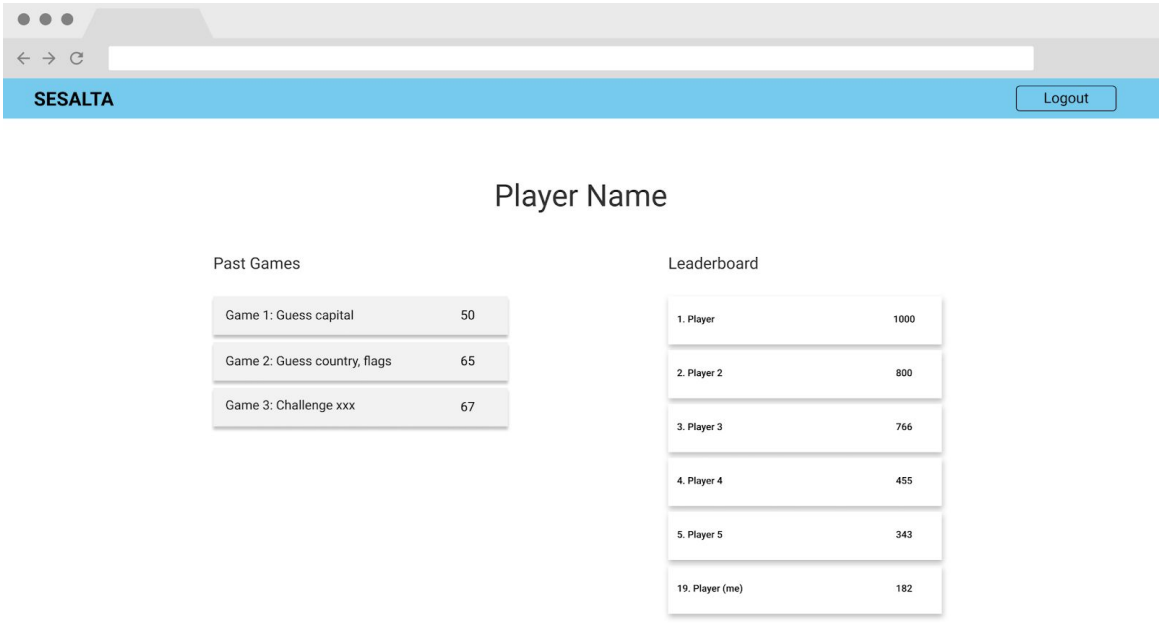
Therefore, a Japanese language option is expected to be added. This acts as both a practical extension to the game as well as a proof of concept.

Additional languages can then be added based on demand and feasibility.

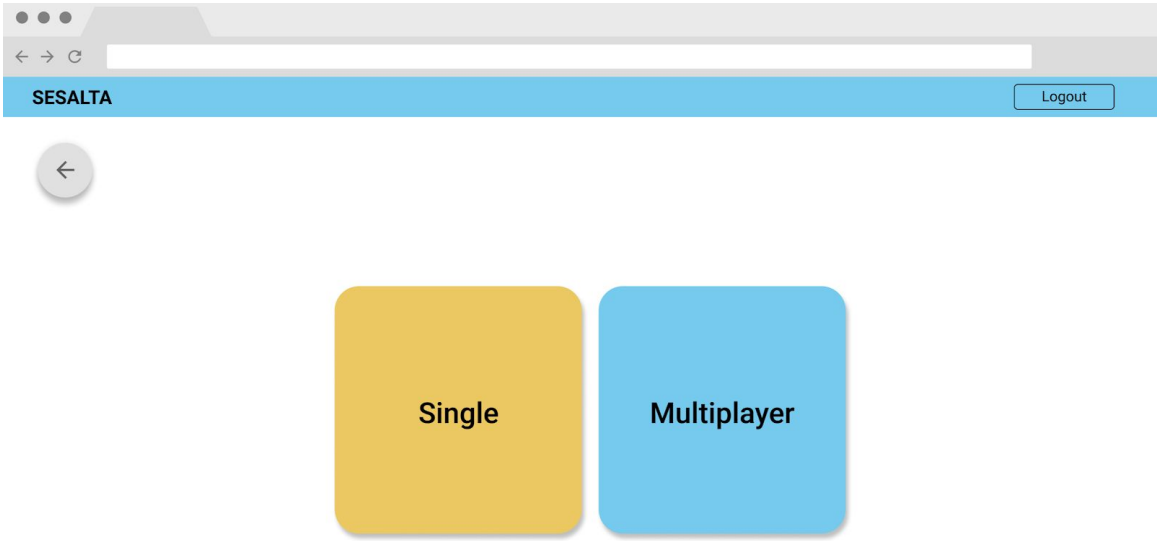
Visual Design



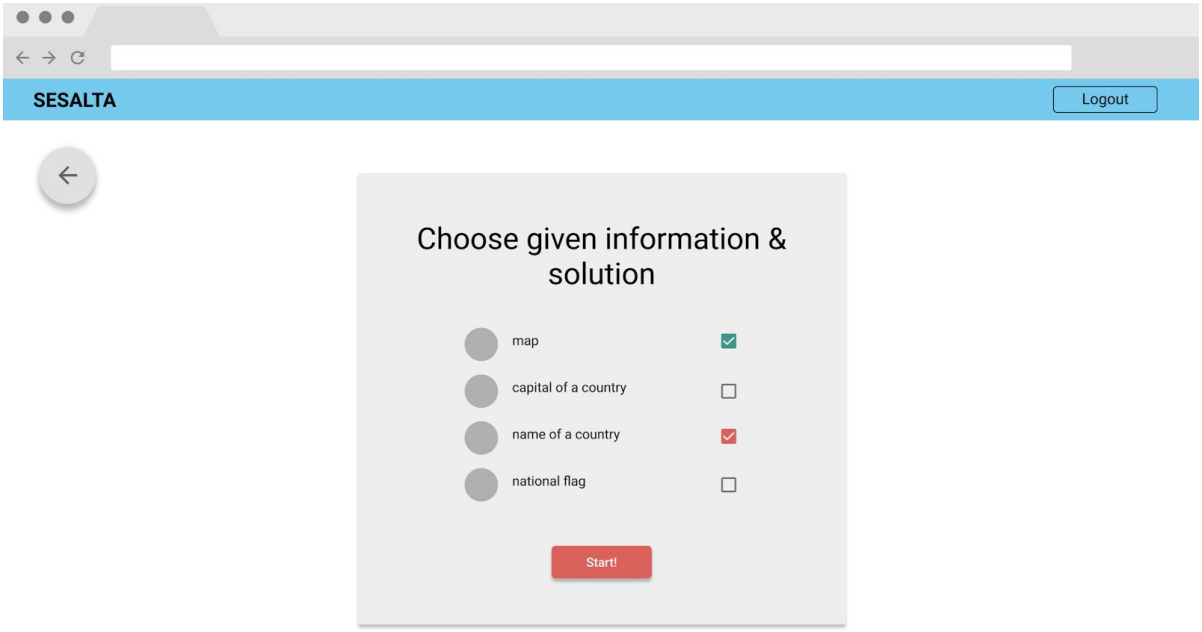
Home page



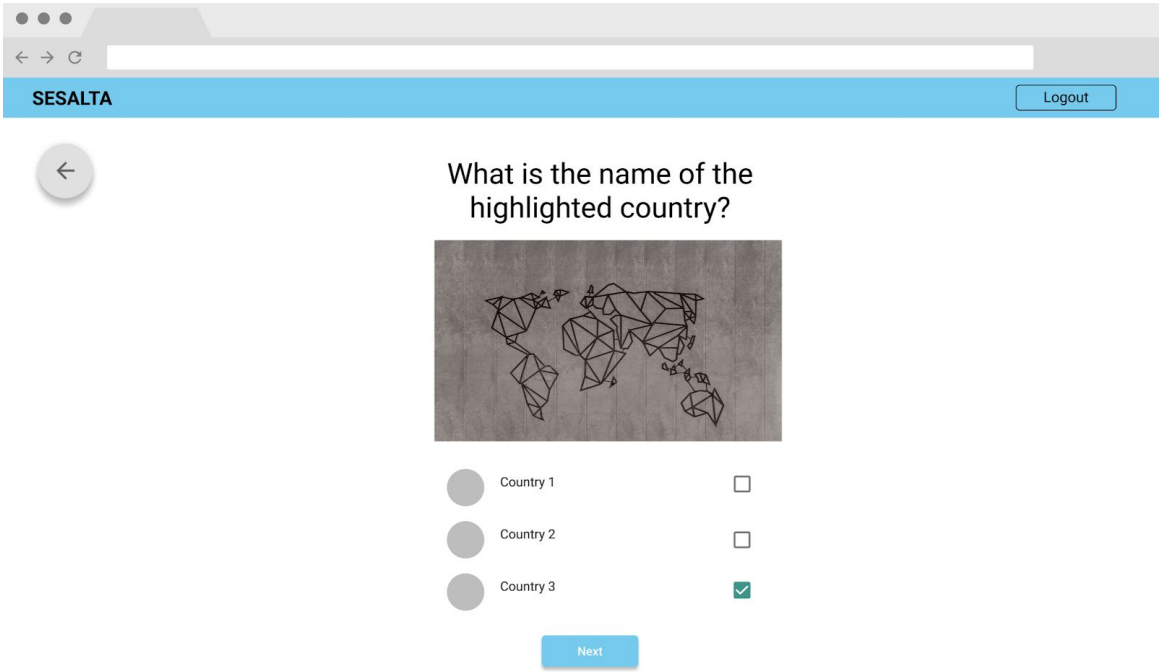
Dashboard



Choose game mode



Choose game options



Play the game

Project Management Tools

Project Management: We use Pivotal tracker[1] for project management. It has a shared backlog and makes priorities clear so our team can stay organized. It also provides guided iteration planning to help the team break down and prioritise projects into manageable chunks.

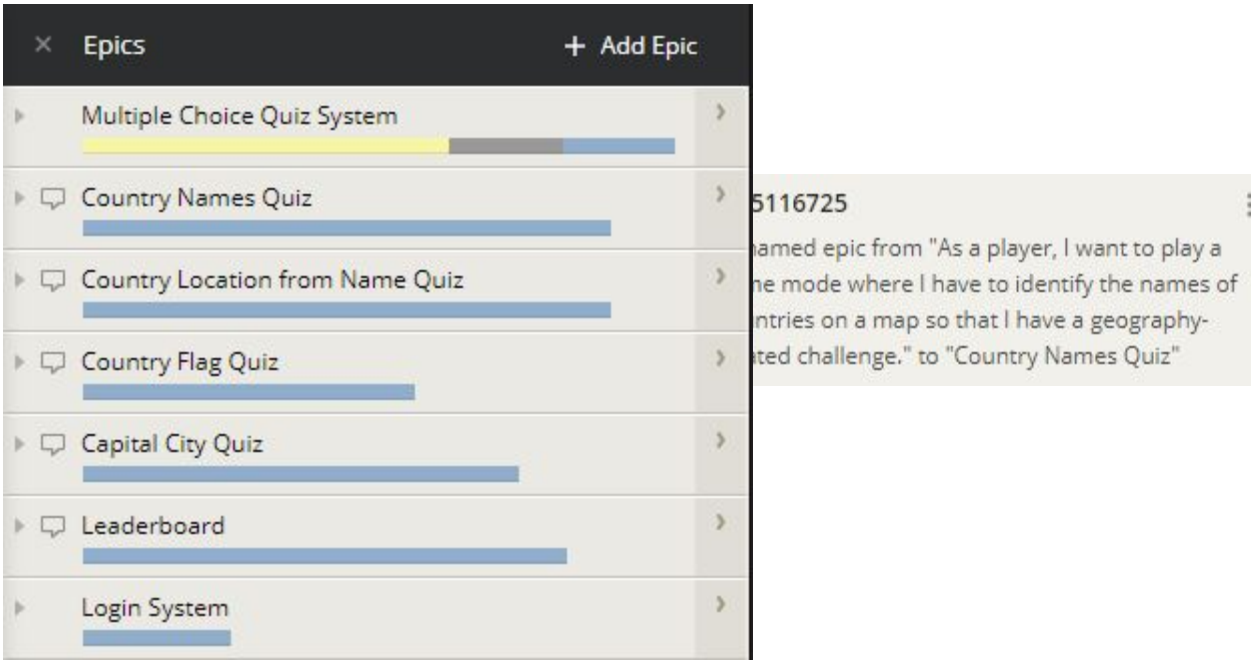
Version Control: We use Git[2] and Github[3] for version control. Git provides the flexibility to work on different feature branches when priorities change. Github has a user-friendly interface and allows project managers and developers to collaborate, track and update work in one place.

Communication: We use Facebook Messenger[4] for daily team communication as it's free and easy to use.

[1] https://www.pivotaltracker.com/
[2] https://git-scm.com/
[3] https://github.com/
[4] https://www.messenger.com/

User Stories

User stories for this project are defined alongside all other planning aspects, using Pivotal Tracker. Below are the Epic Stories, all of which have a full story description but have simplified titles to increase clarity.

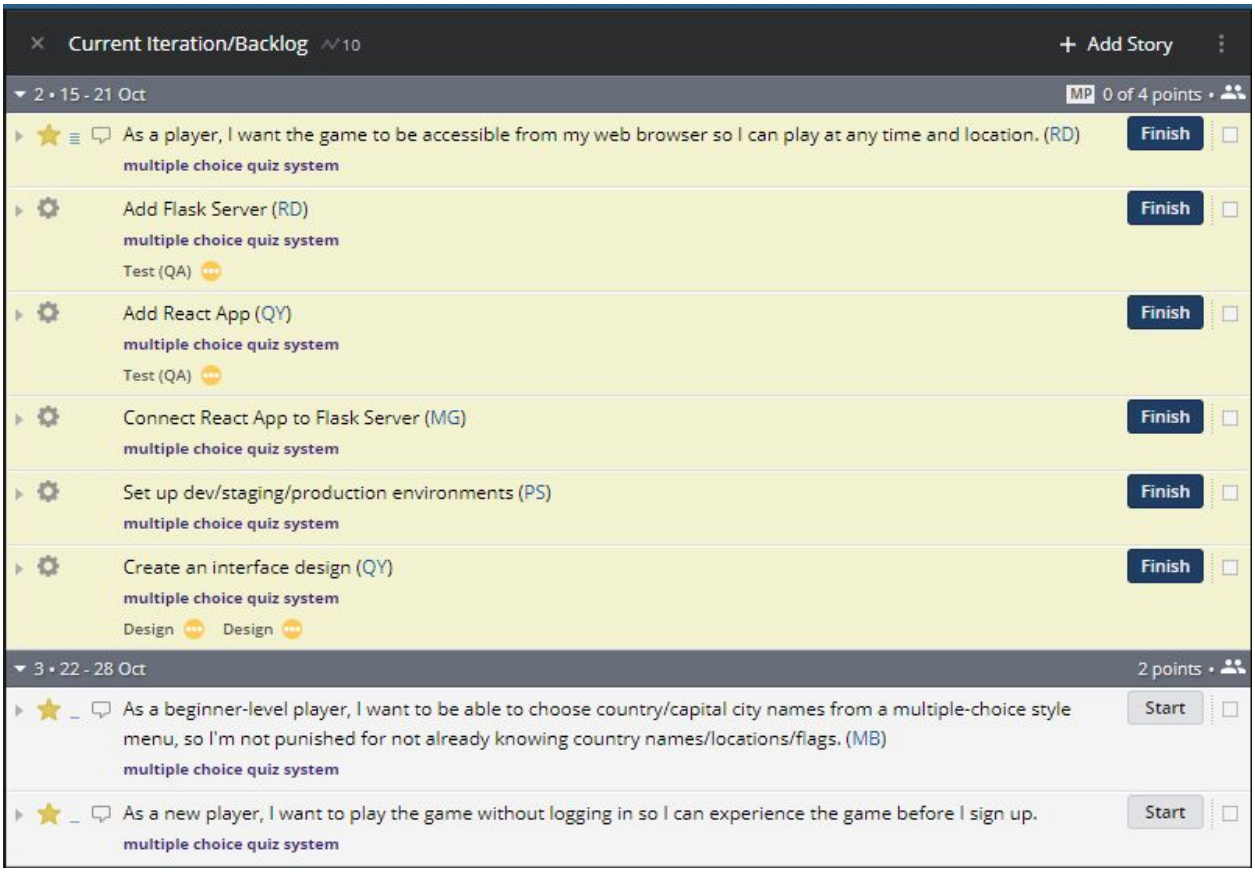


The full list of Epic Stories, and an example of the full Story as a description

Sprint Plan

Next is the current iteration/backlog, which outlines user stories to be completed as well as the tasks required to complete them. All user stories, whether in the current iteration or icebox, have been given point estimations in a 1/2/4/8 system, which was chosen as the most intuitive method of determining story importance/scale.

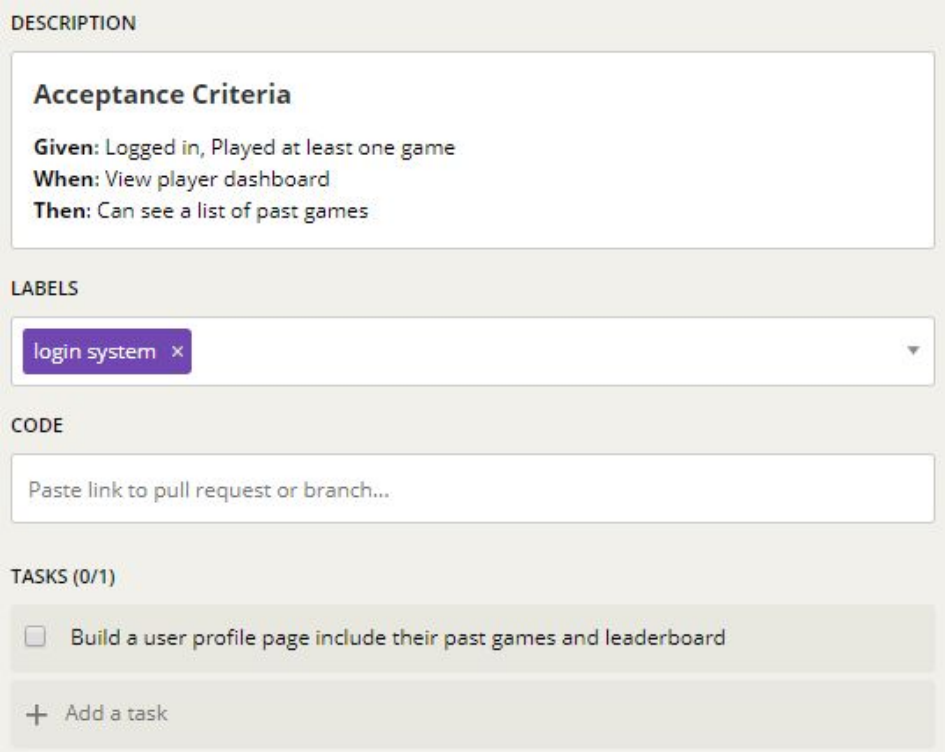
Each task in this section has been assigned group members to work on the story, as well as at least one other member who will test the completed task.



The current iteration/backlog at the time of writing this report

Finally is the Icebox, which contains the list of remaining User Stories, in a similar format with Acceptance Criteria outlined in their descriptions, points allocated using 1/2/4/8, and Tasks listed.

The Icebox, featuring the list of remaining user stories



An example of the Acceptance Criteria, tasks, and how Stories are sorted under each Epic using labels

Release Plan

Details of the minimum viable product and additional features are given in the Product Outline section of this document. This MVP constitutes the initial release of Sesalta.

Release Version	Expected Release Date	Release Log
1.0	28 October, 2019	
2.0	TBA	The second release will contain additional game modes and a new language option (Japanese).
3.0	TBA	The third release will introduce a more complex scoring system and scoreboards (both local and global).
4.0	TBA Only if time permits.	Difficulty settings, trophies and login system.

Meeting Schedule

Meeting	Date	Time
Daily Stand-up	Monday, Wednesday, Thursday, Friday, Saturday[5]	10:00 AM
Sprint Grooming	Sunday	6:00 PM
Sprint Retro	Tuesday	2:00 PM
Sprint Review	Tuesday	2:30 PM
Sprint Planning	Tuesday	3:00 PM

[5] Note: No Stand-Up required on Sunday and Tuesday as updates will be given during the Sprint Meetings scheduled on those days.

Technologies

Front-End Stack

ReactJS: React is a library for building composable user interfaces. It encourages the creation of reusable UI components, which present data that changes over time.[6]

Material-UI: Material-UI provides React components for faster and easier web development.[7]

Babel: Babel is a toolchain that is mainly used to convert ECMAScript 2015+ code into a backwards compatible version of JavaScript in current and older browsers or environments. [8]

Webpack: webpack is used to compile JavaScript modules.[9]

Yarn: Yarn is a package manager for our code.[10]

[6] https://reactjs.org/blog/2013/06/05/why-react.html
[7] https://material-ui.com/
[8] https://babeljs.io/docs/en/
[9] https://webpack.js.org/guides/getting-started/
[10] https://yarnpkg.com/en/docs/getting-started

Back-End Stack

Flask: Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications.[11]

Flask-RESTPlus: Flask-RESTPlus is an extension for Flask that adds support for quickly building REST APIs.[12]

MySQL: MySQL is a relational database management system. It will be incorporated in version 3.0 to facilitate global scoreboards.[13]

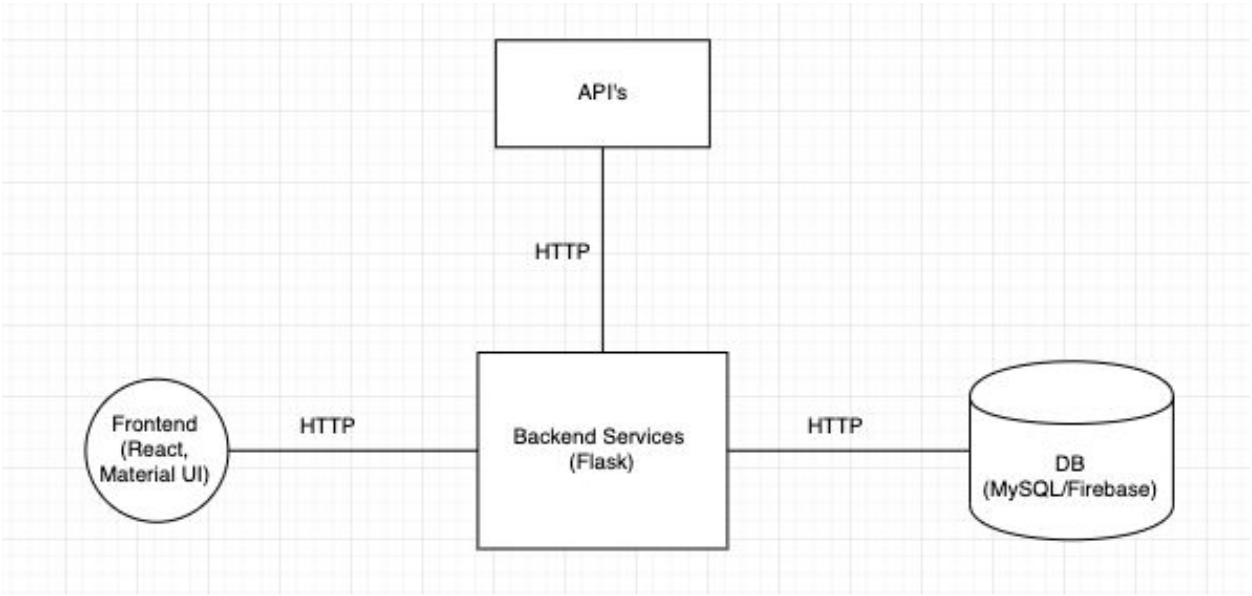
Deployment

Heroku: Heroku is a platform as a service (PaaS) that enables developers to build, run, and operate applications entirely in the cloud.[14]

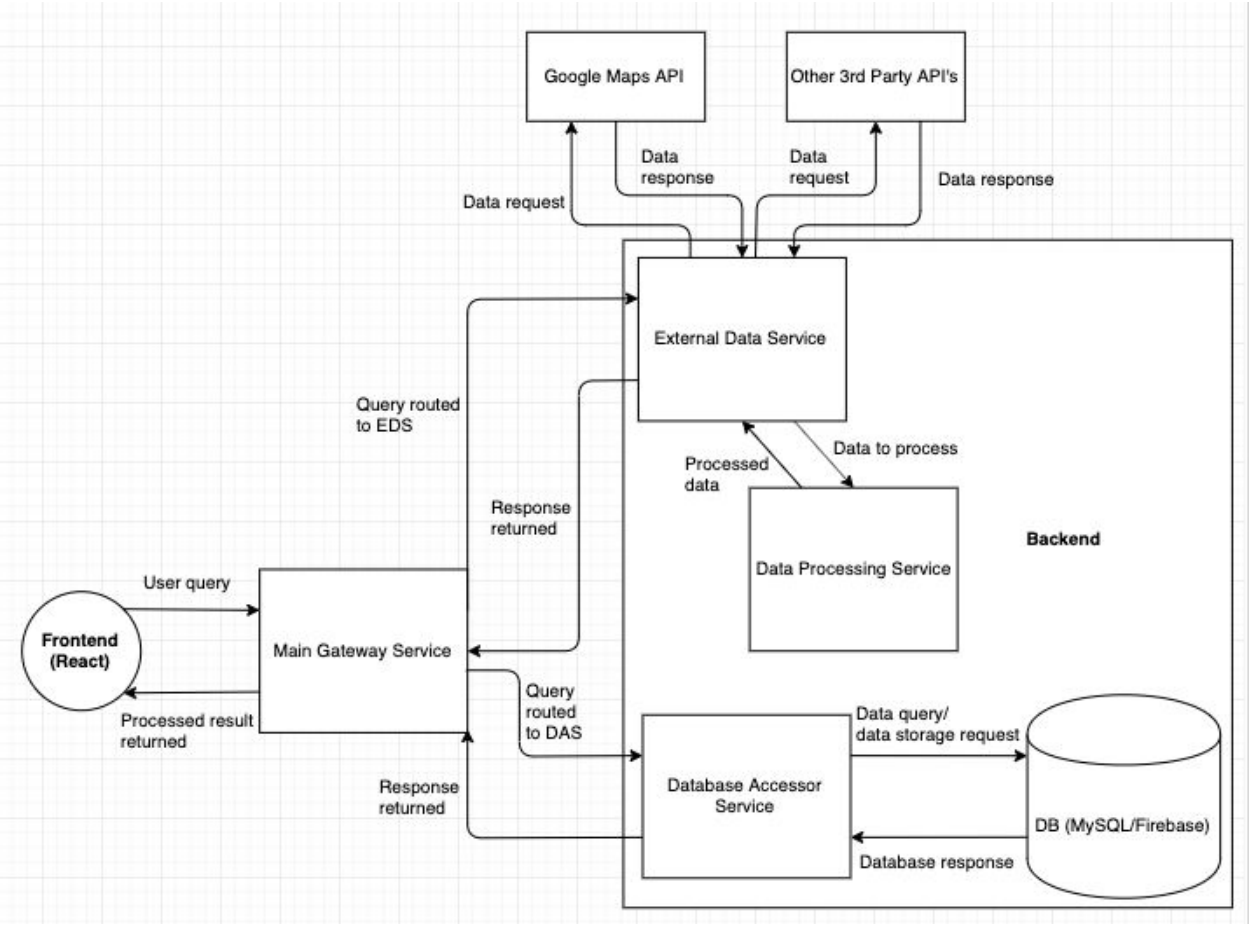
CircleCI: CircleCI automates our software builds, tests, and deployments.[15]

[11] https://palletsprojects.com/p/flask/
[12] https://flask-restplus.readthedocs.io/en/stable/
[13] https://www.mysql.com/
[14] https://www.heroku.com/
[15] https://circleci.com/docs/2.0/about-circleci/

System Architecture and Data Flow



The overall system architecture is segmented into four modular components. A Frontend written in the React framework with the help of Material UI will be communicating via HTTP with our Backend Services. These Backend Services will be written in Python and served using Flask. The Backend Services will communicate with other APIs via HTTP to retrieve any external data required. All user and game data will be stored in our MySQL/Firebase database and is only accessible through our Backend Services.



This is an extended view of our architecture which includes the theoretical dataflow. A good system architecture has modular components each with their own responsibility and this is what we have decided to incorporate into our system.

The Main Gateway Service is the router for all Frontend requests. There is no processing in this service and will only be passing data traffic through it.

The Database Accessor Service is the only component that is capable of connecting with the database and retrieving the data.

The External Data Service retrieves data from other APIs when requested. If the data retrieved requires time consuming processing, it is given to the Data Processing Service to process.

The Data Processing Service will be implemented such that it is capable of processing data requests from any micro-service. Currently it only processes data provided by the External Data Service though we might find during our implementation that this service can be better utilised through interactions with our other services.

All our components have their own responsibilities which allows the system to be scalable. A scalable system will allow us to code our project without having to worry about large amounts of traffic from users breaking parts of the code.

Team Roles

Product Owner (Matthew) - Will put forward his vision for the product and convey it to the scrum team. He helps to prioritise the product backlog. He will also understand the users needs, the market, the competition and any future trends. Finally, he is in charge of making the final decision on any of the concepts brought forward.

Scrum Master (Raycole) - Will manage the project by keeping the team on track for weekly sprint deliveries and helping the team reach a consensus on weekly sprint tasks.

Developers (Everyone) - All team members will be developing the product including the Product Owner and Scrum Master. We have assigned further roles to our team members to efficiently complete this project. Qiwei and Mitchell will spend the majority of their time developing the Frontend for this project and Raycole, Prasad and Matthew will be in charge of developing the Backend.

Relevant Extreme Programming Practices

Pair Programming

Drawing on the varied experiences of the members of the team, we will use Pair Programming to increase knowledge share and help less experienced members on the team make a meaningful contribution to the project.

We plan to split people up as follows:
Front-End: Qiwei has significant experience with React, so he will pair with Mitchell and teach him the basics.

Back-End: Raycole/Matt have experience with Flask, so they will pair and consolidate their existing knowledge

Database: Raycole/Prasad have both done Database Systems and worked with databases before, so they will pair and setup/maintain our database.

Continuous Integration

The development team should always be working off the latest version of master. This can be achieved by rebasing off of master every day. This will reduce the potential merge conflicts later on and avoid any delays in the project cycle.

Coding Standards

To keep our code-base in a healthy state and reduce the time it takes for code reviews, we will enforce a consistent style and format for source code. We will use the following tools to achieve this:

Flake8: Enforces the PEP8 style guide on Python code.

Black: Formats Python code automatically, adheres to PEP8 rules and results in a smaller diff for code reviews.[16]

ESLint: Enforces a set of rules by identifying problematic patterns in JavaScript code.[17]

Prettier: Formats JavaScript code and enforces a consistent code style.

To support code style and formatting, we will also be using type-checkers/type hints to ensure that the code is self-documenting as much as possible. We will use the following tools to achieve this:

TypeScript: A typed superset of JavaScript, and includes its own compiler. Being a typed language, TypeScript can catch errors and bugs at build time.[18]

Mypy: Optional static type checker for Python that aims to combine the benefits of dynamic typing and static typing.[19]

Test-Driven Development

To make sure that our code is working at each iteration, we will use a TDD approach where we write unit tests to test the functionality of each piece of code. Not only will this reduce the time taken during code reviews to test the code, but it will identify issues in the code during the development phase. To adopt this approach, we will use the following tools:

Pytest: The pytest framework makes it easy to write small tests, and also scales to support complex functional testing for applications and libraries.[20]

Jest: Jest allows for easy mocking and snapshot testing of React components.[21]

---

[16] https://github.com/psf/black
[17] https://github.com/eslint/eslint
[18] https://reactjs.org/docs/static-type-checking.html
[19] http://mypy-lang.org/
[20] https://docs.pytest.org/en/latest/
[21] https://jestjs.io/docs/en/tutorial-react

References

1. Neil Harrison, Frances Bodkin, Gawaian Bodkin-Andrews, Elizabeth Mackinlay. (2017) Sensational pedagogies: Learning to be affected by country. *Curriculum Inquiry* 47:5, pages 504-519. (https://doi.org/10.1080/10382046.2016.1262509)