**Requirement Gathering and Analysis Phase**

**Technology Stack (Architecture & Stack)**

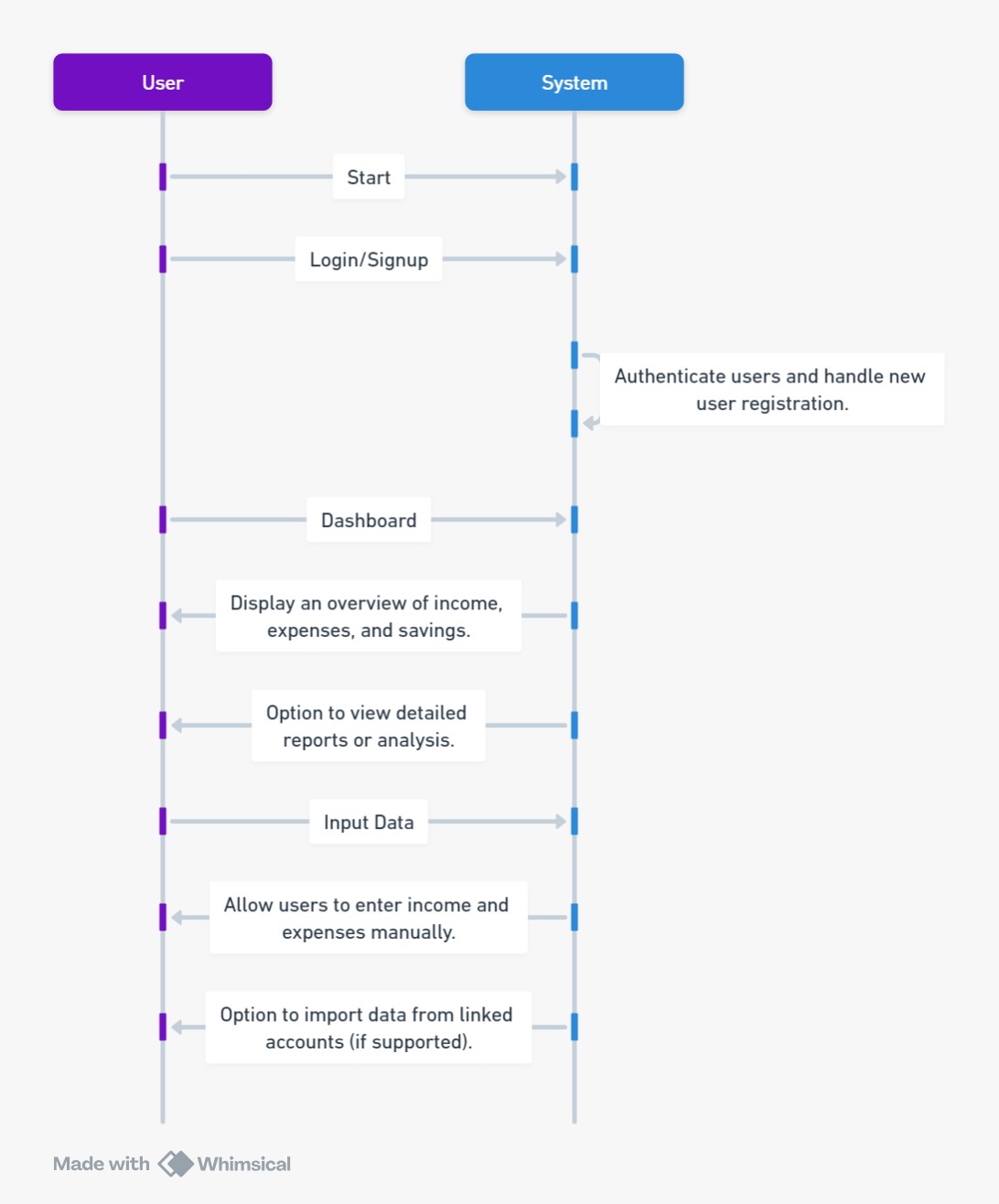
|  |  |
| --- | --- |
| Date | 06-july-2024 |
| Team ID | SWTID1719978597 |
| Project Name | Project – WalletWatch-Expense tracker app |
| Maximum Marks |  |

**Technical Architecture:**

**Reference:**

**https://www.cprime.com/resources/blog/how-to-build-a-money-management-app-requirements-features**

<https://www.researchgate.net/publication/269846460_Design_and_Implementation_Money_Management_Web_Based_Application_for_Personal_and_Family_Proposed_for_CV_X>



**Table-1 : Components & Technologies:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
|  | User Interface | The user gets the detailed analysis and pictorial representations of the expenses vs income. The user can manually enter the data. | HTML, CSS, JavaScript ,React Js |
|  | Document Data Model | It stores data in JSON, BSON, or XML documents. in this data model, we can move documents under one document and apart from this, any particular elements can be indexed to run queries faster. | MongoDB |
|  | Database | MongoDB stores objects in a binary format called BSON. BinData is a BSON data type for a binary byte array | MongoDB |
|  | File Storage | Objects should be less that 16MB. | MongoDB |
|  | Atlas Data API | Purpose of External API used in the application | Data API connects to MongoDB Atlas and is a managed service that lets you securely work with data stored in Atlas using standard HTTPS requests |
|  | Infrastructure (Server / Cloud) | a minimum of 2 CPU cores and 2 GB of RAM for MongoDB Agent hosts. | MongoDB Atlas |

**Table-2: Application Characteristics:**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | Open-Source Frameworks | List the open-source frameworks used | Technology of Opensource framework |
|  | Security Implementations | Authentication using user id and password  Authentication using gmail. | OAuth |
|  | Scalable Architecture | Scaling in MongoDB is a critical process that ensures a database can handle increasing data volumes, user traffic and processing demands. As applications grow, maintaining optimal performance and resource utilization becomes essential. | MongoDB |
|  | Availability | The MongoDb database linked to the app can provide access to the user at any time of the day | MongoDB atlas |
|  | Performance | 10,000 concurrent requests to avoid HTTP response status code of 429 | MongoDB |

**References:**

[**https://medium.com/analytics-vidhya/adding-sign-in-with-google-to-your-website-b82755b79b31**](https://medium.com/analytics-vidhya/adding-sign-in-with-google-to-your-website-b82755b79b31)

[**https://www.mongodb.com/docs/atlas/app-services/reference/service-limitations/#:~:text=10%2C000%20concurrent%20requests.,5%2C000%20concurrent%20Device%20Sync%20connections**](https://www.mongodb.com/docs/atlas/app-services/reference/service-limitations/#:~:text=10%2C000%20concurrent%20requests.,5%2C000%20concurrent%20Device%20Sync%20connections)**.**

[**https://medium.com/distributed-knowledge/scalable-web-architectures-concepts-design-6fd372ee4541**](https://medium.com/distributed-knowledge/scalable-web-architectures-concepts-design-6fd372ee4541)