**# Expense Reimbursement System (ERS) API**

The Expense Reimbursement System (ERS) will manage the process of reimbursing employees for expenses incurred while on company time. All employees in the company can login and submit requests for reimbursement and view their past tickets and pending requests. Finance managers can log in and view all reimbursement requests and past history for all employees in the company. Finance managers are authorized to approve and deny requests for expense reimbursement.

**# Models**

**~~\*\*User\*\*~~**

~~The User model keeps track of users information.~~

~~```javascript~~

~~{~~

~~userId: number, // primary key~~

~~username: string, // not null, unique~~

~~password: string, // not null~~

~~firstName: string, // not null~~

~~lastName: string, // not null~~

~~email: string, // not null~~

~~role: Role // not null~~

~~}~~

~~```~~

**~~\*\*Role\*\*~~**

~~The Role model is used to track what permissions a user has~~

~~```javascript~~

~~{~~

~~roleId: number, // primary key~~

~~role: string // not null, unique~~

~~}~~

~~```~~

**~~\*\*Reimbursement\*\*~~**

~~The Reimbursement model is used to represent a single reimbursement that an employee would submit~~

~~```javascript~~

~~{~~

~~reimbursementId: number, // primary key~~

~~author: number, // foreign key -> User, not null~~

~~amount: number, // not null~~

~~dateSubmitted: number, // not null~~

~~dateResolved: number,~~

~~description: string, // not null~~

~~resolver: number, // foreign key -> User~~

~~status: number, // foreign ey -> ReimbursementStatus, not null~~

~~type: number // foreign key -> ReimbursementType~~

~~}~~

~~```~~

**~~\*\*ReimbursementStatus\*\*~~**

~~The ReimbursementStatus model is used to track the status of reimbursements. Status possibilities are `Pending`, `Approved`, or `Denied`.~~

~~```javascript~~

~~{~~

~~statusId: number, // primary key~~

~~status: string // not null, unique~~

~~}~~

~~```~~

**~~\*\*ReimbursementType\*\*~~**

~~The ReimbursementType model is used to track what kind of reimbursement is being submitted. Type possibilities are `Lodging`, `Travel`, `Food`, or `Other`.~~

~~```javascript~~

~~{~~

~~typeId: number, // primary key~~

~~type: string, // not null, unique~~

~~}~~

~~```~~

**# Endpoints**

**~~## Security~~**

~~Security should be handled through session storage.~~

~~Admin should have access to everything by default. If an endpoint specifies admin that means only admin should be allowed.~~

~~If a user does not have permission to access a particular endpoint it should return the following:~~

~~\*~~ **~~\*\*Status Code:\*\*~~** ~~401 UNAUTHORIZED <br />~~

**~~\*\*Content:\*\*~~**

~~```javascript~~

~~{~~

~~"message": "You are not authorized to access this functionality/area. Please contact your local administrator."~~

~~}~~

~~```~~

~~Occurs if they do not have the appropriate permissions.~~

**## Available Endpoints**

~~[POST /login](#login)~~

~~[GET /users](#find-users)~~

~~[GET /users/:id](#find-users-by-id)~~

~~[PATCH /users](#update-user)~~

[GET /reimbursements/status/:statusId](#find-reimbursements-by-status)

[GET /reimbursements/author/userId:userId](#find-reimbursements-by-user)

~~[POST /reimbursements](#submit-reimbursement)~~

[PATCH /reimbursements](#update-reimbursement)

**~~### \*\*Login\*\*~~**

~~\*~~ **~~\*\*URL\*\*~~**

~~`/login`~~

~~\*~~ **~~\*\*Method:\*\*~~**

~~`POST`~~

~~\*~~ **~~\*\*Request:\*\*~~**

~~```javascript~~

~~{~~

~~username: string,~~

~~password: string~~

~~}~~

~~```~~

~~\*~~ **~~\*\*Response:\*\*~~**

~~```javascript~~

~~User~~

~~```~~

~~\*~~ **~~\*\*Error Response\*\*~~**

~~\*~~ **~~\*\*Status Code:\*\*~~** ~~400 BAD REQUEST~~

~~```javascript~~

~~{~~

~~message: "Invalid Credentials"~~

~~}~~

~~```~~

**~~### \*\*Find Users\*\*~~**

~~\*~~ **~~\*\*URL\*\*~~**

~~`/users`~~

~~\*~~ **~~\*\*Method:\*\*~~**

~~`GET`~~

~~\*~~ **~~\*\*Allowed Roles\*\*~~** ~~`finance-manager`~~

~~\*~~ **~~\*\*Response:\*\*~~**

~~```javascript~~

~~[~~

~~User~~

~~]~~

~~```~~

**~~### \*\*Find Users By Id\*\*~~**

~~\*~~ **~~\*\*URL\*\*~~**

~~`/users/:id`~~

~~\*~~ **~~\*\*Method:\*\*~~**

~~`GET`~~

~~\*~~ **~~\*\*Allowed Roles\*\*~~** ~~`finance-manager` or if the id provided matches the id of the current user~~

~~\*~~ **~~\*\*Response:\*\*~~**

~~```javascript~~

~~[~~

~~User~~

~~]~~

~~```~~

**~~### \*\*Update User\*\*~~**

~~\*~~ **~~\*\*URL\*\*~~**

~~`/users`~~

~~\*~~ **~~\*\*Method:\*\*~~**

~~`PATCH`~~

~~\*~~ **~~\*\*Allowed Roles\*\*~~** ~~`admin`~~

~~\*~~ **~~\*\*Request\*\*~~**

~~The userId must be presen as well as all fields to update, any field left undefined will not be updated.~~

~~```javascript~~

~~User~~

~~```~~

~~\*~~ **~~\*\*Response:\*\*~~**

~~```javascript~~

~~User~~

~~```~~

**### \*\*Find Reimbursements By Status\*\***

Reimbursements should be ordered by date

\* **\*\*URL\*\***

`/reimbursements/status/:statusId`

For a challenge you could do this instead:

`/reimbursements/status/:statudId/date-submitted?start=:startDate&end=:endDate`

\* **\*\*Method:\*\***

`GET`

\* **\*\*Allowed Roles\*\*** `finance-manager`

\* **\*\*Response:\*\***

```javascript

[

Reimbursement

]

```

**### \*\*Find Reimbursements By User\*\***

Reimbursements should be ordered by date

\* **\*\*URL\*\***

`/reimbursements/author/userId/:userId`

For a challenge you could do this instead:

`/reimbursements/author/userId/:userId/date-submitted?start=:startDate&end=:endDate`

\* **\*\*Method:\*\***

`GET`

\* **\*\*Allowed Roles\*\*** `finance-manager` or if ther userId is the user making the request.

\* **\*\*Response:\*\***

```javascript

[

Reimbursement

]

```

**### \*\*Submit Reimbursement\*\***

\* **\*\*URL\*\***

`/reimbursements`

\* **\*\*Method:\*\***

`POST`

\* **\*\*Rquest:\*\***

The reimbursementId should be 0

```javascript

Reimbursement

```

\* **\*\*Response:\*\***

\* **\*\*Status Code\*\*** 201 CREATED

```javascript

Reimbursement

```

**### \*\*Update Reimbursement\*\***

\* **\*\*URL\*\***

`/reimbursements`

\* **\*\*Method:\*\***

`PATCH`

\* **\*\*Allowed Roles\*\*** `finance-manager`

\* **\*\*Request\*\***

The reimbursementId must be presen as well as all fields to update, any field left undefined will not be updated. This can be used to approve and deny.

```javascript

Reimbursement

```

\* **\*\*Response:\*\***

```javascript

Reimbursement

```

**# Stretch Goals**

These are not part of the core requirements but are things that could be worked on once the core requirements are done.

\* Password Hashing

\* Paging ans Sorting endpoints: [Reference For How](https://docs.microsoft.com/en-us/azure/architecture/best-practices/api-design#filter-and-paginate-data)

\* Using JSON Web Tokens (JWTs) instead of Session Storage

\* Being able to submit receipts. (I would recommend using AWS S3 buckets for this but if you do be cautious of including AWS Access Keys in your application)