```
            ...mapGetters('auth', [ 'companies']),

            previousRoute() {
                return {
                    name: 'recruitment-posting',
                    params: {
                        workforce_id: this.workforce_id,
                        job_posting_pk: this.$route.params.job_posting_pk
                    }
                };
            }
        },

        async created() {
            if (!this.$route?.query?.applicant_id) {
                this.$router.replace({
                    name: 'recruitment-posting',
                    params: this.$route.params
                });
                return;
            }
            this.getJobOfferApplicantDetails(
                parseInt(this.$route.query.applicant_id))
                .then(response => {
                    this.applicant = response;
                    return this.fetchScheduleGroups();
                })
                .then(() => {})
                .catch(e => {
                    console.error('Error in getJobOfferApplicantDetails',
e);
                });
        },

        methods: {
            ...mapActions('jobOffer', ['getJobOfferApplicantDetails',
'createJobOffer']),
            ...mapActions('contract', ['createContractPolicy']),
            ...mapActions('employers', ['updateOrCreateEmploymentRule',
'fetchJobScheduleGroups']),
```

```
            };

            if (jobOffer.jobDescription) {
                data.job_description_id = jobOffer.jobDescription.pk;
            }

            return this.createJobOffer(data)
                .then(response => {
                    if (response && response.pk) {
                        this.$router.push({
                            name: 'recruitment-job-offer',
                            params: {
                                ...this.$route.params,
                                job_offer_id: response.pk
                            }
                        });
                    } else {
                        let msg = `Sorry, error encountered while
creating Job offer.
                        Kindly try again.`;

                        this.errorMessage = response.error ?
response.error : msg;
                    }
                }).catch(e => {
                    console.error('Error in createJobOffer', e);
                }).finally(() => {
                    this.isSubmitting = false;
                    return data;
                });
        },

        getContractPolicyPk(params) {
            let contractPolicyPk;
            this.createContractPolicy(params)
                .then((response) => {
                    contractPolicyPk = response.pk;
                })
                .catch((e) => {
```

```
                    this.errorMessage = 'Sorry, there was an error
creating a Contract Policy';
                    console.error(e);
                });
            return contractPolicyPk;
        },

        getJobPositionValue(position) {
            if (position === 'Manager') {
                return MANAGER;
            } else if (position === 'Supervisor') {
                return SUPERVISOR;
            } else {
                return RANK_AND_FILE;
            }
        },

        getEmployeeTypeValue(type) {
            if (type === 'Exempt') {
                return EXEMPT;
            } else {
                return NONEXEMPT;
            }
        },

        getCompensationTypeValue(type) {
            if (type === 'Wage') {
                return HOURLY;
            } else {
                return MONTHLY;
            }
        },

        async getSalaryValue(jobOffer) {
            let value = 0;
            value = parseFloat(jobOffer.compensationValue) ;

            return value.toFixed(6);
        }
    }
```